Návrh a implementace informačního systému pro restaurační zařízení s využitím Java technologií

Design and implementation of an information system for businesses from the field of gastronomy using the Java platform

Bc. Zdeněk Huspenina

Diplomová práce
2008

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav automatizace a řídicí techniky
akademický rok: 2007/2008

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Zdeněk HUSPENINA**
Studijní program:	**N 3902 Inženýrská informatika**
Studijní obor:	**Automatické řízení a informatika**
Téma práce:	**Návrh a implementace informačního systému pro restaurační zařízení s využitím Java technologií**

Zásady pro vypracování:

1. **Proveďte teoretický rozbor alternativ vývojových platforem a nástrojů pro restaurační informační systém.**
2. **Navrhněte informační systém s grafickým uživatelským rozhraním pro restaurační zařízení, ve kterém bude implementován modul autentifikace uživatele a modul objednávkového procesu.**
3. **Uvedený systém realizujte s tím, že technické řešení informačního systému bude postaveno na Java platformě.**
4. **Datové úložiště celého systému vyřešte pomocí běžně používané komerční nebo opensource databázové technologie.**
5. **Zhodnoťte navržený informační systém a navrhněte případné další rozšíření pro praktické využití v komerčním prostředí.**

Rozsah práce:

Rozsah příloh:

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:

1. **Ian F. Darwin: Java – Kuchařka programátora, CP Books a.s., ISBN: 80-251-0944-5**
2. **Brett Spell: Java Programujeme profesionálně, CP Books a.s., ISBN: 80-7226-667-5**
3. **Bogdan Kiszka: 1001 tipů a triků pro programování v jazyce Java, CP Books a.s., ISBN: 80-7226-989-5**
4. **Sharon Zakhour, Scott Hommel, Jacob Royal, Isaac Rabinovitch, Tom Risser, Mark Hoeber: Java 6 – Výukový kurz, CP Books a.s., ISBN: 978-80-251-1575-6**
5. **Steve McConnell: Dokonalý kód -Umění programování a techniky tvorby software, CP Books a.s., ISBN: 80-2510849-X**
6. **Petr Paleta.: Co programátory ve škole neučí aneb Softwarové inženýrství v praxi, CP Books a.s. 2003, ISBN: 80-251-0073-1**
7. **http://developers.sun.com**

Vedoucí diplomové práce: **Ing. Milan Navrátil, Ph.D.**
Ústav elektrotechniky a měření

Datum zadání diplomové práce: **22. února 2008**

Termín odevzdání diplomové práce: **6. června 2008**

Ve Zlíně dne 22. února 2008

L.S.

UNIVERZITA TOMÁŠE BATI VE ZLÍNĚ
Fakulta aplikované informatiky

prof. Ing. Vladimír Vašek, CSc.
děkan

prof. Ing. Vladimír Vašek, CSc.
ředitel ústavu

ABSTRAKT

Rozvíjející se oblast obchodu a služeb s sebou nese nutnost využití všech dostupných nástrojů pro vylepšení stávajících firemních procesů. Právě informační technologie jsou významnou součástí pro vývoj takových nástrojů. Tato práce se zabývá návrhem a implementací informačního systému pro běžné restaurační zařízení se zaměřením na objednávkový proces. V části návrhu jsou popsány alternativy architektur, technologií, nástrojů a datových úložišť pro vývoj řešeného systému. Výsledná aplikace obsahuje funkčnosti, jako jsou autentifikace unikátního uživatele, pořizování, prohlížení objednávek a jejich položek, tisk objednávek, modifikace databázových číselníků a další. Využitím moderních informačních technologií, jako je kombinace Java platformy a Oracle databáze, se stává vhodným prostředkem k naplnění daných cílů.

Klíčová slova: software, Java, databáze, objednávky, restaurace, objektově orientované programování

ABSTRACT

Expanding business and services carry necessity to use all available tools for improvement of current corporate processes. Information technologies represent significant component for development of those tools. This diploma thesis deals with proposal and implementation of information system for common restaurant with a view to order process. In the design part, there are described alternatives of architectures, tochnologies, tools and data storage sites which are essential for solved system development. Final application contains functionalities such as autentification of unique user, making, browsing and printing of orders and their items, modifications of database code lists etc. The application becomes suitable tool for fulfillment of given objectives due to using of modern information technologies such as combination of Java platform and Oracle database.

Keywords: software, Java, database, orders, restaurant, object-oriented programming

Děkuji vedoucímu své diplomové práce Ing. Milanu Navrátilovi, PhD., za odborné vedení, podnětné připomínky a rady udílené při vypracování této práce.

Prohlašuji, že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval.V případě publikace výsledků, je-li to uvolněno na základě licenční smlouvy, budu uveden jako spoluautor.

Ve Zlíně 6. 6. 2008 …….

Podpis diplomanta

OBSAH

ÚVOD

V dnešní době dochází k neustálému rozvoji obchodu a služeb. Avšak s tímto rozvojem rostou i nároky na jednotlivé podnikatele a živnostníky a případný tlak ze strany konkurence. Jestliže chce firma obstát v tomto silném konkurenčním prostředí, musí využít všechny dostupné nástroje a metodiky. Především pomocí správného použití těchto nástrojů lze významně ovlivnit firemní procesy a infrastrukturu daného podniku a vytvořit si tak cennou konkurenční výhodu na daném trhu. Moderní doba firmám nabízí širokou škálu informačních technologií a jejich využití. A právě proto se vhodný informační systém stává plnohodnotným nástrojem pro aktivní rozvoj firmy.

Do úvahy se bere implementace informačního systému v prostředí běžného restauračního zařízení. Pro každé takové zařízení je důležitý objednávkový proces a s ním související aktuální stav objednávek a přehled jejich historie. Při návrhu softwarové aplikace je velmi důležité stanovit její architekturu, protože nevhodně navržená architektura může negativně ovlivnit samotný vývoj celého systému, neboť její pozdější změny jsou velmi komplikované nebo téměř nemožné. Základními alternativami jsou architektury tenkého nebo tlustého klienta. Tenký klient se skládá z klientské, datové a aplikační vrstvy, přičemž klientská vrstva je realizována internetovým prohlížečem a slouží pouze jako rozhraní pro zobrazení grafického uživatelského rozhraní. Tlustý klient se skládá ze dvou vrstev, a to z klientské a datové. Aplikační logika i rozhraní zobrazení GUI je uloženo přímo na klientské stanici. Pro správný návrh všech částí architektury je možné použít CASE nástroje, respektive modelovací jazyk UML.

Daná architektura úzce souvisí s použitými technologiemi. Před samotným vývojem je třeba zvážit možnosti jednotlivých technologií a na základě detailního rozboru pak určit výslednou vývojovou platformu a programovací jazyk, jež budou použity. V současnosti je nejrozšířenější platforma JAVA a .NET, přičemž obě platformy nabízejí podobné vlastnosti a produkty pro vývoj sofistikovaných informačních systémů. Java však nabízí možnost běhu programu na více operačních systémech, a právě z tohoto důvodu bude použita Java platforma. Princip řešeného systému je založen na zpracování a sběru dat, proto musí využít příslušnou databázovou technologii. Výběr datového úložiště závisí na zkušenostech a finančních prostředcích daného vývojového týmu a nabízejí se dvě základní možnosti. Jednou z možností je použít opensource databázi, která je obvykle zdarma nebo některou z komerčních databází, za jejichž použití se platí nemalé licenční poplatky. Pro

vývoj restauračního informačního systému bude použita bezplatná komerční databáze Oracle 10g Express Edition. Řešený informační systém bude určen pro více uživatelů a bude rozdělen na veřejné a privátní sekce, přičemž každý uživatel bude mít přidělena přístupová práva pro vstup do privátních sekcí. Struktura celého systému bude rozdělena na unikátní moduly a submoduly s možností jejich vzájemné interakce. Každý z těchto modulů bude obsahovat příslušnou aplikační logiku a společně pak budou tvořit ucelený koncept objednávkového proces.

I. TEORETICKÁ ČÁST

1 PODMÍNKY REALIZACE A CÍLE

Řešení informačního systému bude navrženo pro desktopový nebo přenosný osobní počítač s operačním systémem Microsoft Windows XP Home (Professional) nebo s libovolnou linuxovou distribucí. Hlavními požadavky informačního systému bude navrhnout intuitivní a uživatelsky přívětivé uživatelské rozhraní s ergonomickým ovládáním. Řešený informační systém bude implementovat modul autentifikace uživatele a modul objednávkového procesu, jenž bude obsahovat aplikační logiku běžného objednávkového procesu v restauračním zařízení.

Technické řešení restauračního informačního systému bude postaveno na Java platformě, přičemž jako datové úložiště bude použita komerční nebo opensource databáze.

1.1 Obecný princip objednávkového procesu

Obecným principem objednávkového procesu v příslušném restauračním zařízení je objednávka nabízených produktů určitým zákazníkem. Objednávkový proces začíná v okamžiku výběru produktů z jídelníčku. Zákazník nahlásí obsluze restaurace své požadavky, obsluha pak zapíše všechny data do informačního systému. V této chvíli dochází k důkladnému uložení všech údajů v databázi, které můžou později sloužit managementu restaurace k prohlížení. Poté přinese obsluha zákazníkovi jeho objednávku. Pokud již zákazník nemá žádné další požadavky, dochází k procesu placení a následnému tisku objednávky s příslušnými položkami objednávky.

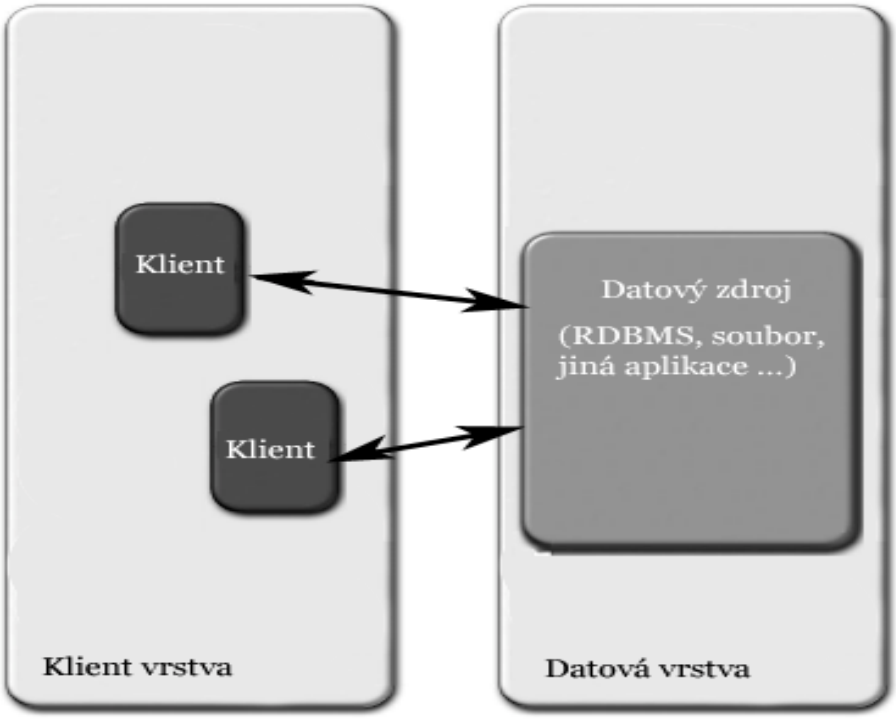
2 ARCHITEKTURA VÝVOJE

Při návrhu a vývoji softwarového produktu je potřeba stanovit na jaké architektuře bude daná aplikace vyvíjena. Architektura je podle všeobecného označení jádro, určující celkovou strukturu a základní konstrukci částí nebo kompletního počítačového systému.

V tomto speciálním případě architektury aplikace se jedná o způsob rozdělení aplikace, aplikačních dat, procesů i datových toků do logických celků, stanovení struktury těchto komponent, vzájemných vztahů a interakcí mezi nimi, a to na dostatečně obecné úrovni. Je zřejmé, že se k návrhu a volbě architektury musí přistupovat na samém počátku vývoje jakékoliv aplikace, a to s velkou opatrností a rozmyslem. Jakékoliv její pozdější změny jsou totiž velmi komplikované nebo téměř nemožné. Hlavními alternativami vývoje je architektura tenkého a tlustého klienta. Při vlastním návrhu a vývoji restauračního informačního systému jsem využil architekturu tlustého klienta, protože je tato architektura v komerčním prostředí pro aplikace tohoto typu typická.

2.1 Tlustý klient (GUI aplikace)

Tlustý neboli bohatý klient je počítač (klient) v klient-serverové síťové architektuře, který poskytuje bohatou funkcionalitu nezávisle na centrálním serveru. Původně byl v ranných dobách osobních počítačů označovaný pouze jako klient nebo silný klient. Přívlastek tlustý je opakem tenkého z termínu tenký klient, což typicky znamená, že aplikace na klientském stroji. Tlustý klient vyžaduje alespoň periodické pripojení k centrálnímu síťovému serveru. Některé činnosti nicméně dokáže vykonávat i bez připojení (viz. například tlustý mailový klient jako je Microsoft Outlook ve srovnání s tenkým mailovým klientským systémem jako je Hotmail). Tlustý klient je také obvykle charakterizován potřebou instalovat specifickou aplikaci.



Obrázek 1. – Architektura tlustého klienta

2.1.1 Použití tlustého klienta

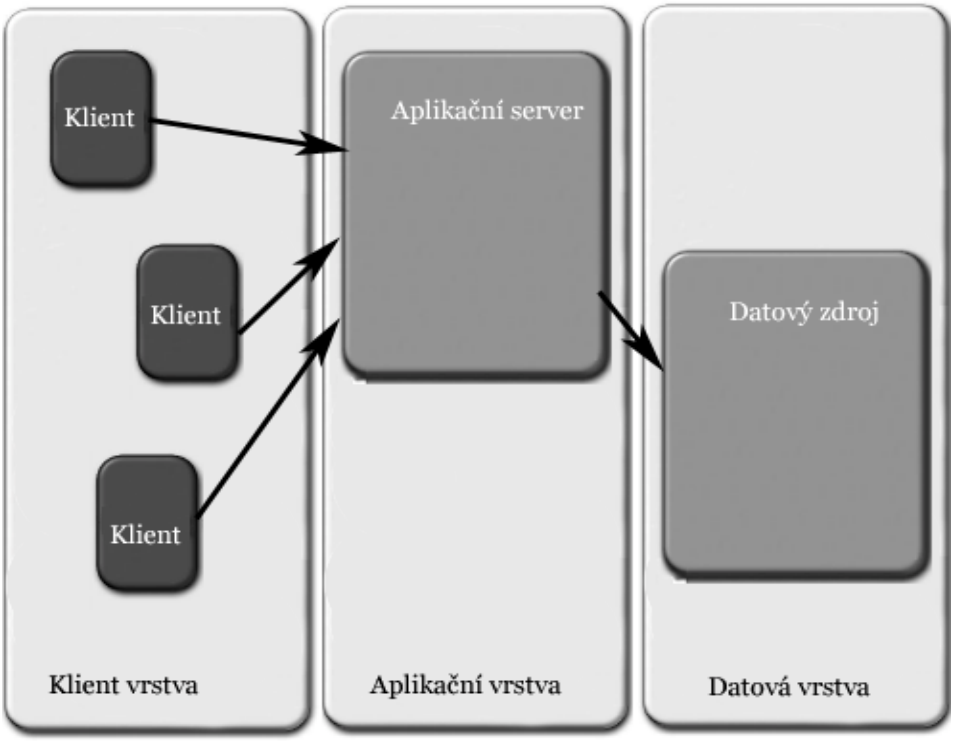
Architektura tlustého klienta se nasazuje zpravidla pro aplikace, u nichž jsou menší nároky na servery, protože tlustý klient nevyžaduje tak vysoký výkon jako tenký klient (tlustí klienti provádějí celou řadu výpočtů sami o sobě). Výsledkem je pak dramaticky levnější

serverová část aplikace. Tlustí klienti mají výhodu v tom, že nevyžadují konstantní připojení k serveru a zprostředkují lepší multimediální výkon. Tlustí klienti mají navrch v multimediálně náročných aplikacích, které by bylo příliš složité obsloužit z hlediska datových přenosů v tenkých klientech.

Softwarové aplikace na některých operačních systémech jsou často navrženy tak, že potřebují vlastní lokální zdroje. Zkoušet rozběhnout takový software v tenkém klientu může být problémová, a proto architektura tlustého klienta poskytuje větší flexibilitu.

2.2 Tenký klient (WEB aplikace)

Tenký klient je aplikace poskytovaná uživatelům z webového serveru přes počítačovou síť Internet, nebo její vnitropodnikovou obdobu (intranet). Tenký klient je realizován prostřednictvím webového prohlížeče, jenž sám o sobě logiku dané aplikace nezná. Aktualizovat a spravovat tenkého klienta bez nutnosti šířit a instalovat software na potenciálně tisíce uživatelských počítačů je hlavním důvodem jejich oblíbenosti. Aplikace typu tenký klient jsou používány pro implementaci mnoha podnikových i jiných informačních systémů. Generují dynamicky sérii webových stránek ve standardním formátu HTML/XHTML, který je podporován běžnými prohlížeči. Obecně je každá jednotlivá webová stránka dodána prohlížeči jako statický dokument, ale sled takových stránek vyvolává interaktivitu, např. díky reakci na vstup uživatele do formulářových prvků vložených do kódu stránky. Pro přidání dynamických prvků do uživatelského rozhraní se obvykle používá skriptovací jazyk JavaScript. Webový prohlížeč v průběhu běhu aplikace interpretuje a zobrazuje stránky a funguje jako univerzální klient pro libovolnou webovou aplikaci.[1]



Obrázek 2. – Architektura tenkého klienta

2.2.1 Rozhraní tenkého klienta

Rozhraní v některých směrech omezuje funkčnost a možnosti klienta. Metody známé z aplikací postavených na architektuře tlustého klienta, jako je například vykreslování na obrazovku či obecné techniky jako drag and drop, nejsou standardními technologiemi prohlížečů podporovány. Pro přidání takovéto funkčnosti se často používají skriptování na straně klienta, zvláště pro vytvoření dojmu interaktivity bez nutnosti znovunačtení stránky, které řada uživatelů shledává rušivým. V poslední době se začínají používat technologie, které umožňují spolupráci skriptů na klientské straně se serverovou částí aplikace. Jedním z příkladů je AJAX (Asynchronní JavaScript a XML), jedná se o techniku vývoje webu využívající kombinaci HTML, JavaScriptu a rozhraní XMLHttpRequest, které umožňuje načítat klientským skriptům informace ze serveru, aniž by bylo třeba obnovovat celou stránku.

2.2.2 Technická specifikace

Podstatnou výhodou vývoje aplikací stavějících na standardních funkcích prohlížeče je jejich schopnost pracovat podle určení bez ohledu na operační systém či jeho verzi instalovanou na daném klientském počítači. V praxi ale nekonzistentní implementace HTML, CSS, DOM a další specifikace jednotlivých prohlížečů způsobují problémy. Navíc mají uživatelé možnost nastavit způsob zobrazení ve svém prohlížeči (např. zvolit jiný řez či velikost písma, barvy či vypnout podporu skriptování), což může rušit jednotný vzhled aplikace. Dalším způsobem, avšak méně častým, je použití Macromedia Flash nebo javových appletů pro část nebo celé uživatelské rozhraní. V současnosti je novou technologií od společnosti Microsoft produkt SilverLight, jenž je velkou konkurencí Macromedia Flash. Zřejmou nevýhodou tohoto přístupu je vysoká závislost na poskytovateli aplikace a dostatečně dimenzované kapacitě připojení k serveru poskytovatele. Jestliže se poskytovatel rozhodne ukončit poskytování této služby nebo ji přeruší z jiného důvodu, nelze službu nadále používat, na rozdíl od lokálně provozovaného software. Stejně tak pokud dojde k přerušení spojení se serverem poskytovatele, může být služba dočasně nedostupná. [1]

3 VÝVOJOVÉ PLATFORMY A NÁSTROJE

Vývojové platformy a použité nástroje jsou velmi důležitým aspektem, jenž může podstatně ovlivnit návrh a vývoj celého informačního systému. V dnešní době máme k dispozici celou řadu těchto platforem a nástrojů, jež zvyšují produktivitu programátorů. Mnoho z těchto nástrojů jsou komerční, tudíž se musí zaplatit, avšak na trhu existuje i několik opensource nástrojů, jež jsou volně ke stažení.

Pro správný návrh architektury aplikace lze využít CASE nástroje, respektive UML jazyk. Mezi další nástroje používané přímo při vývoji patří integrovaná vývojová prostředí (IDE), která se zpravidla skládají z editoru, kompilátoru, debbugeru, plnohodnotné referenční nápovědy a dalších možných přídavných modulů. S počtem přídavných modulů integrovaného prostředí však narůstá složitost prostředí, protože se zvyšuje množství informací, které jsou k programování nezbytné. Volba IDE se obvykle odvíjí z použité platformy, protože každá vývojová platforma má ve svém balíku toto prostředí již integrované. Spoustu práce mohou ušetřit také veřejné balíky knihoven, které mají klasickou formu programových modulů nebo knihovny sloužící pro vizuální rozvržení grafického uživatelského rozhraní celé aplikace. Vývoj s použitím vizuálních knihoven je obvykle rychlejší než návrh struktury uživatelského rozhraní bez jejich použití, avšak v určitém průběhu vývoje se můžou stát černou skříňkou a z tohoto důvodu jsem žádné takové GUI knihovny nepoužil a veškeré uživatelské rozhraní jsem vytvářel manuálně přímo ve zdrojovém kódu.

3.1 Modelovací jazyk UML a CASE nástroje

Modelovací jazyk UML (Unified Modeling Language) je výsledkem snažení analytiků a designérů, kteří v průběhu 80. a 90. let vytvářeli metody, jež by umožnily popsat objektově orientovanou analýzu a návrh. UML se používá v softwarovém inženýrství na vizualizaci, specifikaci, navrhování a dokumentaci programovaných systémů. UML je souhrnem grafických notací k vyjádření analytických a návrhových modelů. Jedinečný modelovací jazyk umožňuje modelovat jednoduché i složité aplikace pomocí stejné formální syntaxe. UML jazyk spadá do množiny CASE nástrojů.

CASE nástroje jsou nástroje pro podporu analýzy a návrhu aplikací (Computer Aided Software Engineering). V současnosti všechny ve světě rozšířené objektově

orientované CASE nástroje vycházejí z modelovacího jazyka UML. Pouze plnohodnotné CASE nástroje ale umožní propojení jednotlivých technik UML, sdílení modelů mezi členy týmu, a tedy týmovou práci. Dále pak CASE nástroje zpravidla obsahují možnosti generování a synchronizace kódů objektových prostředí (např. prostředí .NET, Java, C++ atd.), včetně generování databázových skriptů pro založení databáze, reverzního engineeringu a správy datových modelů. Většina takových nástrojů umožňuje i procesní modelování jako doplněk k technikám UML. CASE produkty jsou ideálním nástrojem pro návrh informačních systémů, protože umožňují v rámci jediného prostředí identifikovat důležité diagramy UML (ale i procesní a datové modely) potřebné pro analýzu a návrh informačních systémů.

Pří vývoji restauračního informačního systému byly využity kromě případů užití také modely tříd objektů, model objektové spolupráce a pro persistentní data datové modelování. [2]

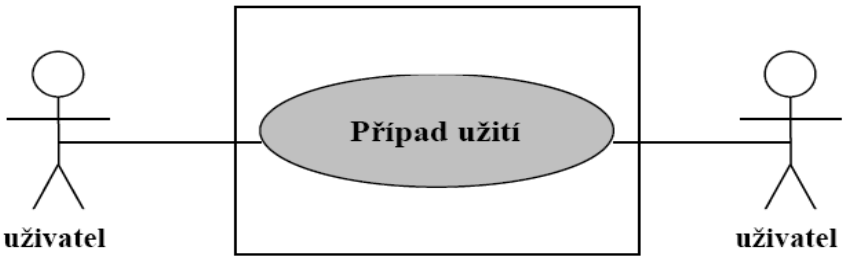
3.1.1 Případy užití

Případy užití zachycují funkčnost celého budoucího informačního systému a zároveň přesně vymezují rozsah prací. Každý případ užití popisuje jeden ze způsobu použití systému, popisuje tedy jednu jeho požadovanou funkčnost. Případ užití je sada scénářů, které spojuje dohromady společný cíl. Příklady případů užití:

- Založení nového zákazníka
- Vytvořit novou objednávku
- Tisk objednávky

3.1.2 Aktéři

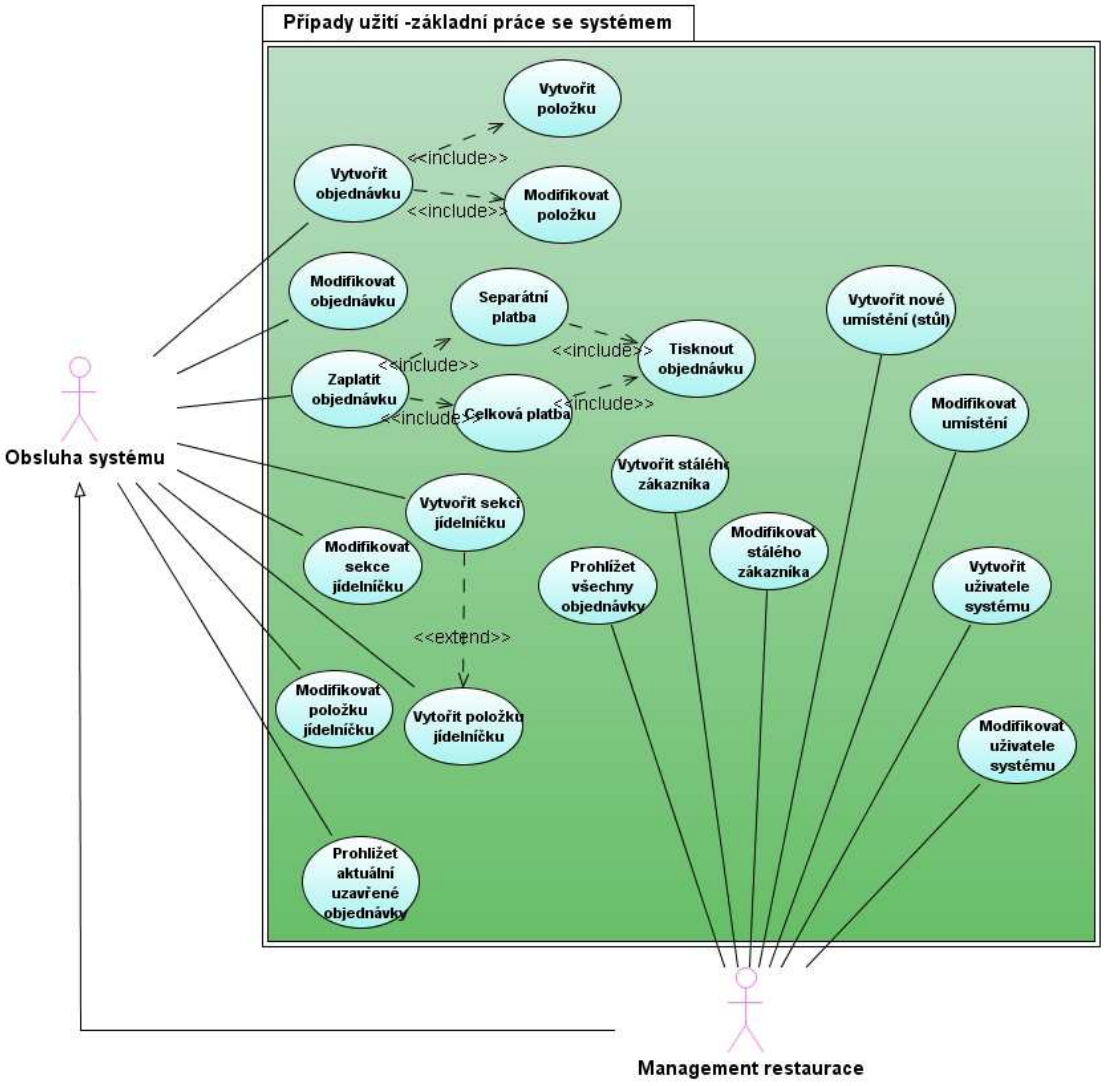
Aktér je uživatel pracující se systémem, přičemž veškeré akce jsou vyvolány právě aktérem. Aktérem může být i externí objekt, který si vyměňuje informace se systémem. Každý aktér provádí určité případy užití. V systému může jeden aktér provádět řadu případů užití a obráceně, jeden případ užití může být prováděn více aktéry.



Obrázek 3. – Vztah mezi aktéry a případy užití

Aktéři v restauračním informačním systému:

- Obsluha restauračního zařízení (číšník/servírka)
- Management podniku - provozní manager podniku



Obrázek 4. – Případy užití – základní práce se systémem

3.2 Alternativy vývojových platforem

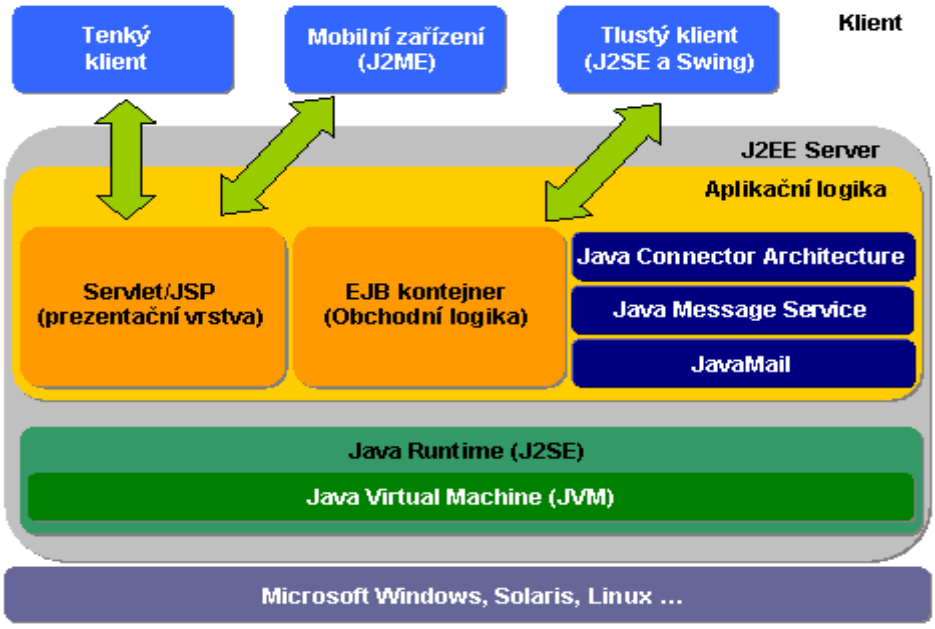
Před samostatným vývojem informačního systému je potřeba zvolit vývojovou platformu, respektive použitý programovací jazyk. V běžné praxi volba probíhá přímo použitím jedné základní vývojové platformy vyvinuté danou softwarovou společností, například Microsoft, Sun Microsystems, IBM a další. Ale v rámci této platformy většinou máme určitou možnost výběru programovacího jazyka. Na platformě Microsoft se například můžeme rozhodnout pro použití klasického C++ nebo některý jazyk z rodiny .NET. U Sunu lze použít například jazyk Java. Každá volba má své pro a proti. Nevhodně zvolený programovací jazyk může zvýšit pracnost projektu. Užší výběr byl tedy stanoven na jazyk Java a jazyk .NET z důvodu jejich největšího rozšíření v oblasti vývoje softwaru. Protože bude systém postaven na architektuře tlustého klienta a z důvodů možnosti běhu programu na více operačních systémech., bude použit programovací jazyk Java.

3.2.1 Vývojová platforma Java

Java je programovací jazyk pocházející od firmy Sun Microsystems. Jedná se o objektově orientovaný jazyk, jehož syntaxe a vlastnosti vycházejí původně z C++, ke kterému má také syntakticky nejblíže. Velkou výhodou Javy je také její hardwarová nezávislost díky virtuálním stroji JVM.

Aplikační programové rozhraní Javy neboli API Javy je předem připravený kód, jenž je uspořádán do tématicky jednotných balíčků, které slouží pro specifický softwarový vývoj. Rozhraní API Javy se dělí na následující tři platformy:

- **Java Micro Edition (J2ME)** – Tato sada poskytuje optimalizované prostředí vývoje a následného zpracování určené pro spotřebitele produktů, jako jsou mobilní telefony, pagery a systémy navádění automobilu.
- **Java Standard Edition (J2SE) -** Tato platforma obsahuje základní sadu tříd v jazyku Java a třídy potřebné pro tvorbu grafického uživatelského rozhraní.
- **Java Enterprise Edition (J2EE)** – Tato platforma v sobě obsahuje API třídy a knihovny J2SE a API určené zpravidla pro vývoj webových aplikací a serverů.



Obrázek 5. – Aplikační rozhraní Javy

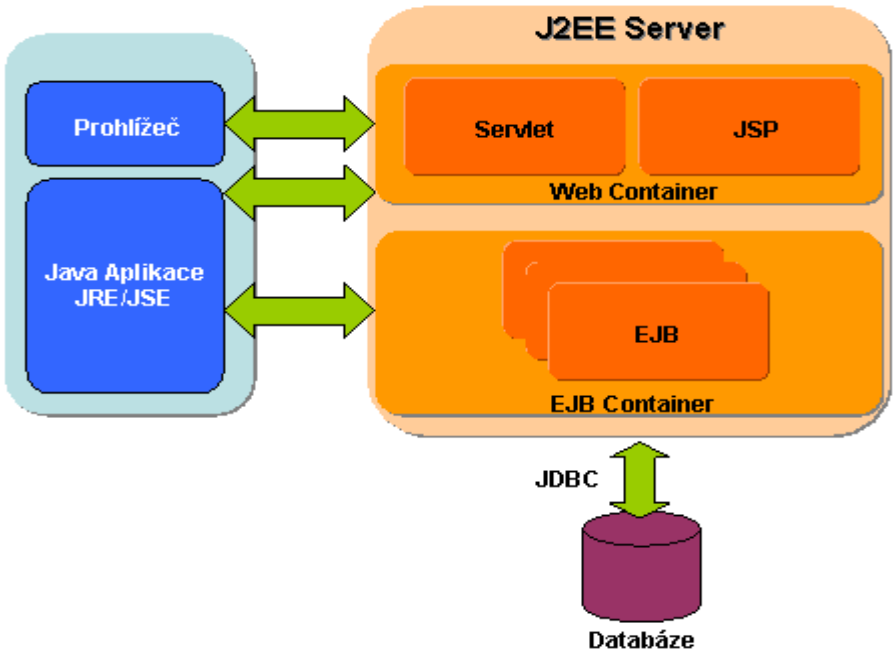
3.2.1.1 Virtuální stroj jazyka Java (JVM)

Jedná se o software, který vystupuje jako hardwarové zařízení a umožňuje spouštění programů psaných v jazyce Java, převádí jejich bajtový kód do nativních instrukcí pro hostitelský počítač. Jiné programovací jazyky jako C, C++ využívají překladač, jenž je závislý nejen na typu procesoru, ale nezřídka i na operačním systému. Překladač tak překládá zdrojový kód na kód spustitelný. Výsledný spustitelný soubor je pak soběstačnou aplikací, kterou lze spouštět na daném typu počítače. Jejich velkou nevýhodou je ovšem skutečnost, že takovéto aplikace nelze přenášet do jiných prostředí. U Javy není výstupem z překladače standardní spustitelný kód. Překladač jazyka totiž generuje optimalizovanou sadu instrukcí nazývanou bajtovým kódem. Tato sada instrukcí je pak program, jenž je interpretován právě virtuálním strojem jazyka Java. Jestliže je tedy na jakékoli cílové stanici nainstalována JVM, lze pak tuto aplikaci bez problémů spustit.Spuštěná aplikace je pak vlastní instancí virtuálního stroje.

3.2.1.2 Java 2 Enterprise Edition

J2EE neboli Java Enterprise Edition je platforma, která zpravidla jako jeden termín označuje celý balík technologií. Architektura J2EE je základem integrace všech typů aplikací. Ať již při vývoji tenkého či tlustého klienta, klient-server aplikace nebo n-

úrovňové systémy. J2EE určuje jak tyto aplikace vyvíjet, jak je provázat a jaké technologie se mají použít. Základem celé architektury je J2SE, neboli Java Standard Edition, která nenabízí žádné takové technologie jako J2EE, ale jedná se o základní funkční API (dalo by se částečně přirovnat k tak zvanému operačnímu systému Javy). J2SE může mít formát obvykle jen čistého runtimu JRE (Java Runtime Enviroment), nebo může obsahovat i vývojové knihovny, tzv. JDK (Java Development Kit). Nad touto základnou se pak staví další aplikační služby nebo tento runtime sloužící pro běh aplikace. Níže uvedený diagram popisuje J2EE platformu, skládající se z aplikačních komponent (sem jsou zahrnuty klientské aplikace, applety, servlety, JSP a EJB) a kontejnerů (poskytující runtime pro manažery jednotlivých systémových zdrojů). Aplikační komponenty jsou právě ty části, s kterými přijde vývojář přímo do kontaktu a které i sám vyvíjí.



Obrázek 6. – J2EE platforma a její komponenty

Hlavní váha J2EE spočívá v serverové i klientské části. Především snadnost a efektivnost využití jednotlivých technologií nabízí vývojářům odstínění od mnoha systémových problémů, které by je jinak zcela zbytečně zpomalovaly. Díky úspoře času v plánování a definování úkolů, kdy se lze díky vlastnostem v této platformě soustředit primárně na aplikační úlohy. Kromě toho jsou API navržená tak, že se s nimi lze snadno naučit pracovat. [3]

Základní součásti J2EE:

- **JDBC (Java Database Connectivity) -** Jedná se o API umožňující unifikovaný přístup k libovolnému RDBMS, který nabízí odpovídající JDBC ovladač pro připojení. Dalo by se říci, že JDBC nabízí podobnou funkčnost jako ODBC, ale s mnoha vylepšeními, kdy je především nutné zmínit objektový model přístupu k databázi
- **JSP (JavaServer Pages) -** JSP jsou určeny pro generování dynamického obsahu pro webové prohlížeče a také pro mobilní zařízení. Nejčastěji se JSP používají ke generování HTML stránek zobrazujících výsledky dotazů uživatelů.
- **JSF (JavaServer Faces) -** JSF je velmi pravděpodobně reakcí na inovativní prvky, které přinesl .NET. JSF jsou mladá technologie a zatím ve fázi specifikačního procesu. Jedná se o komponenty generující uživatelské rozhraní na straně serveru bez ohledu na výsledný protokol, což výrazně usnadňuje a urychluje vývoj aplikací.
- **Servlety -** Servlety jsou základem JSP, a to proto, že JSP jsou po kompilaci pregenerovány do servletu (to vše řeší JSP runtime automaticky). Servlety se užívají hlavně na rychlé a optimalizované zpracování dotazů, kdy ve velkých systémech primárně slouží k rozhazování dat a dotazů na EJB v aplikačním serveru.
- **RMI (Remote Method Invocation) -** Jedná se o protokol definující postupy, jak vyvolávat metody objektů, které nejsou vytvořeny v rámci jednoho procesu. V podstatě se jedná o interprocesní komunikaci, kdy kód obvykle běží v jiném procesu a na jiném počítači. Proto RMI definuje architekturu vzdálených objektů a síťové komunikace a způsob přenášení dat mezi těmito vzdálenými objekty.

3.2.2 .NET Framework

.NET je zastřešující název pro soubor technologií v softwarových produktech, které tvoří celou platformu, která je dostupná nejen pro Web, Windows, ale i Pocket PC.

Základní komponentou je Microsoft .NET Framework, prostředí potřebné pro běh aplikací a nabízející jak spouštěcí rozhraní, tak potřebné knihovny.

- Microsoft .NET Framework - je nejrozšířenější platforma pro osobní počítače s operačním systémem Microsoft Windows od verze Windows 98.
- Microsoft .NET Compact Framework - je platforma určená pro kapesní počítače a mobilní telefony s operačním systémem Windows Mobile.
- Microsoft .NET Micro Framework - je platforma určená pro embedded zařízení, s ještě menší výpočetní kapacitou a většími omezeními, než kapesní počítače.

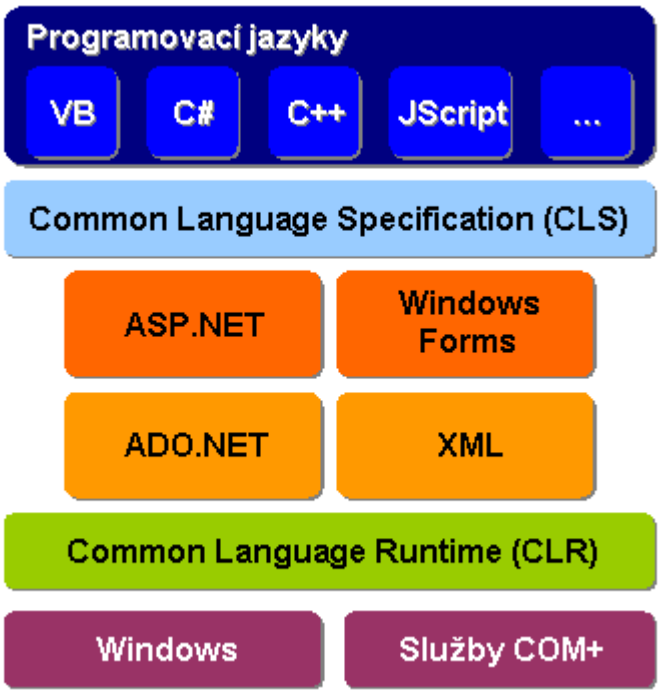
Platforma Microsoft .NET se v základní podobě skládá ze čtyř hlavních částí. První je .NET Framework tvořící spolu s MS Visual Studiem .NET ucelené vývojové prostředí pro tvorbu webových XML služeb. Jádrem celé platformy je právě sada programovacích rozhraní .NET Framework, vedle celé řady tříd obsahuje také obecný jazykový runtime, který řídí provádění kódu napsaného v libovolném podporovaném programovacím jazyce, a ASP.NET. .NET implementuje rozhraní umožňujících tvorbu aplikací pro mobilní inteligentní zařízení. MS Visual Studio .NET je pokračováním úspěšného vývojového balíku, opět nabízejícího podporu více jazyků (Visual Basic, C++, nový jazyk C# a JScript), v rámci platformy .NET lze ale využít také další programovací jazyky od třetích stran, jako je například COBOL či Perl. Jazyků, které je možno bez větších problémů zařadit do architektury .NET, jsou asi čtyři desítky.

Součástí platformy jsou také tzv. stavební bloky služeb – uživatelsky zaměřených webových XML služeb. Tyto služby vlastně tvoří jeden ze základů funkčnosti celé platformy .NET. Některé bloky přitom poskytne přímo Microsoft (ověřování totožnosti, zasílání zpráv, personalizace apod.), další pak partneři a nezávislí vývojáři. Pro přístup k jednotlivým XML službám uživatelé využívají prostředků nazývaných .NET experiences, které jsou vlastně kombinací samotných XML služeb s případným lokálním aplikačním kódem. Třetí hlavní částí platformy Microsoft .NET jsou serverová prostředí – k nim

vedle samotných operačních systémů patří například Application Center podporující provoz škálovatelných aplikací, Biz Talk Server 2000 určený pro vývoj a řízení obchodních procesů založených na XML, Mobile Information Server zpřístupňující vytvořené aplikace mobilním zařízením a v neposlední řadě také databázová platforma v současnosti pod označením MS SQL Server 2008.

Poslední částí platformy Microsoft .NET je podpora tzv. inteligentních zařízení – díky ní je možné aplikace provozovat na celé řadě zařízení, od osobních počítačů přes notebooky až po kapesní počítače či dokonce herní konzole. Podstatné přitom je, že v rámci platformy Microsoft .NET lze transparentně využít celou řadu vlastností, jimiž tato zařízení disponují, tedy jako je například jednoznačná identifikace uživatele zařízení včetně personalizace nebo transparentní přístup k datům odkudkoli a kdykoli.

Platforma .NET tedy obsahuje vše, co je potřebné pro vývoj a provoz aplikací – sadu vývojových nástrojů, programovacích rozhraní a stavebních prvků, odpovídajících serverových (jak operačních, tak aplikačních) prostředí a podporu celé řady zařízení. [4]



Obrázek 7. – Jádro platformy .NET Framework

3.3 Integrované vývojové prostředí (IDE)

Vývojové prostředí, je softwarový nástroj usnadňující práci programátorů, většinou zaměřený na jeden konkrétní programovací jazyk. Obsahuje editor zdrojového kódu, kompilátor, případně interpret a většinou také debugger. Některé obsahují systém pro rychlý vývoj aplikací (zvaný RAD), který slouží pro vizuální návrh grafického uživatelského rozhraní. Pokud se jedná o nástroj pro objektově orientované programování, může obsahovat také object browser, což je nástroj pro prohlížení objektů. Mezi nejpoužívanější vývojové prostředky pro programování Java aplikací patří zpravidla opensource projekt NetBeans a projekt Eclipse. Další alternativou je použít integrované vývojové prostředí JBuilder nebo JEdit a další. V konečném důsledku záleží čistě pouze na každém programátorovi, které IDE použije. Pro vývoj restauračního informačního systému bylo použito IDE NetBeans.

3.3.1 NetBeans

NetBeans je open source projekt s rozsáhlou uživatelskou základnou, komunitou vývojářů a s více než 100 partnery po celém světě. Pod firmu Sun Microsystems, která je hlavním sponzorem projektu, přešel na základě akvizice stejnojmenné české společnosti v říjnu 1999. Pod open-source licencí byl produkt uvolněn v červnu 2000.

V rámci projektu existují dva hlavní produkty: vývojové prostředí NetBeans (NetBeans IDE) a vývojová platforma NetBeans (The NetBeans Platform).

Vývojové prostředí NetBeans IDE je nástroj, pomocí něhož lze psát, překládat, ladit a šířit programy. NetBeans IDE je napsáno v jazyce Java a je postaveno na stejnojmenné platformě. Primárně je určeno pro vývoj aplikací v jazyce Java - ale může podporovat i další programovací jazyky (ve verzi 6.0 např. C++, PHP, Ruby). V Javě podporuje všechny 3 hlavní platformy - J2SE, J2EE a J2ME. Zjednodušuje práci s frameworky jako je Struts nebo Hibernate a umožňuje mimo jiné vývoj SOAP aplikací a webových služeb i webových aplikací. Obsahuje také RAD nástroje pro tvorbu designu aplikace. Kromě toho také existuje velké množství modulů, které toto vývojové prostředí rozšiřují. Vývojové prostředí NetBeans je bezplatně šířený produkt, který je možné používat bez jakýchkoliv omezení. Je použitelný na operačních systémech Windows, Linux, Mac OS X a Solaris. Kromě vývojového prostředí je také dostupná vývojová platforma NetBeans Platform, což

je modulární a rozšířitelný základ pro použití při vytváření rozsáhlých aplikací. Nezávislí dodavatelé softwaru poskytují zásuvné moduly pro integraci do této platformy. Tyto moduly slouží pro vývoj jejich vlastních nástrojů a řešení.

Oba produkty jsou vyvíjeny pod open source licencí a je možné je bezplatné používat v komerčním i nekomerčním prostředí. Zdrojový kód je dostupný pod licencí Common Development and Distribution License. [5]

3.3.2 Eclipse

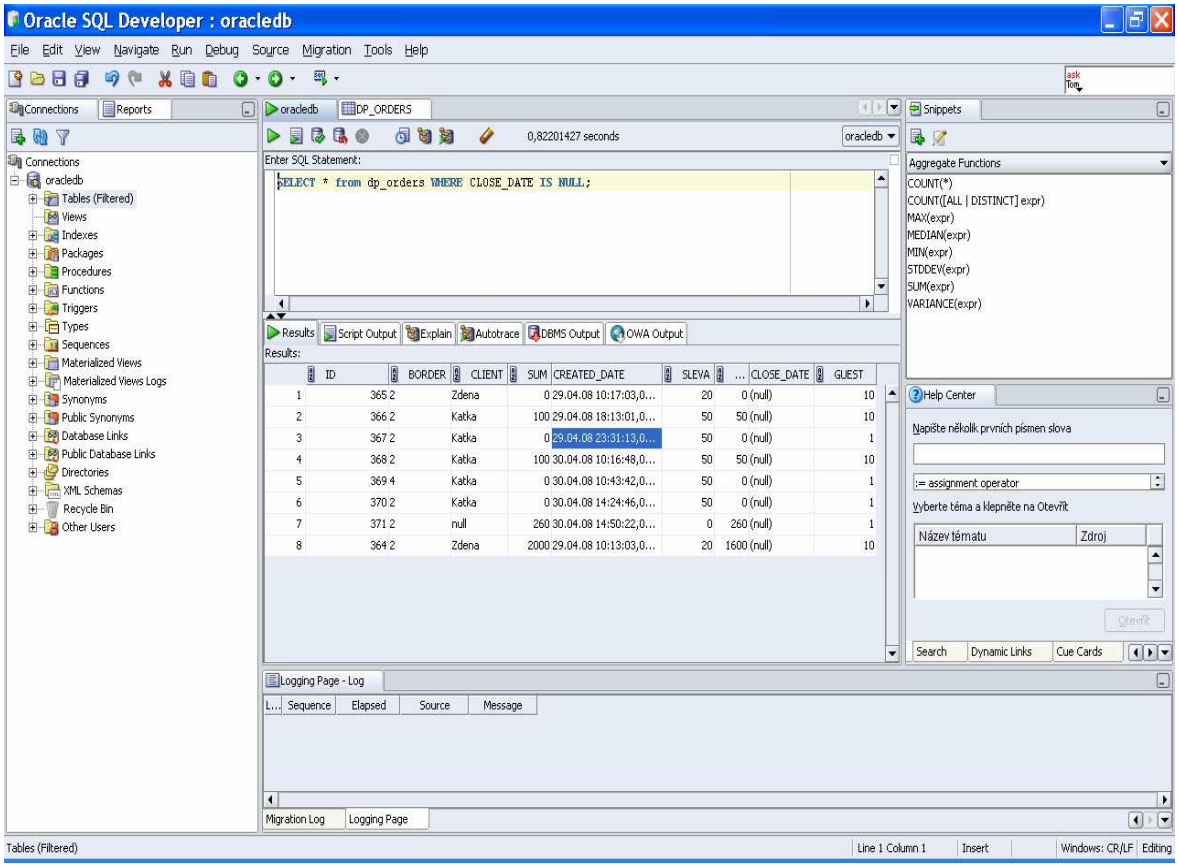
Eclipse je open source vývojová platforma, která je známa jako vývojové prostředí (IDE) určené pro programování v jazyce Java. Flexibilní návrh této platformy dovoluje rozšířit seznam podporovaných programovacích jazyků za pomoci pluginů, například o C++ nebo PHP. Právě pluginy umožňují toto vývojové prostředí rozšířit například o návrh UML, či zápis HTML nebo XML.

Oproti ostatním vývojovým prostředím v Javě, jako například Netbeans, je filozofie Eclipse úzce svázána právě s rozšířitelností pomocí pluginů. V základní verzi obsahuje Eclipse pouze integrované prostředky pro vývoj standardní Javy jako kompilátor, debugger atd., ale neobsahuje například nástroj pro vizuální návrh grafických uživatelských rozhraní desktopových aplikací nebo aplikační server – všechna taková rozšíření je potřeba dodat formou pluginů. Z tohoto důvodu přímo pod křídly Eclipse vznikly takzvané subprojekty, které zastřešují rozšíření pro jednotlivé oblasti softwarového vývoje v Javě. Tyto subprojekty usnadňují integraci potřebných rozšíření do samotného vývojového prostředí.

3.4 Oracle SQL Developer

Oracle SQL Developer je zdarma poskytovaný grafický nástroj pro vývoj a správu v databázi Oracle. Pomocí SQL Developeru je možné vytvářet a prohlížet jednotlivé databázové objekty všech typů a spouštět SQL příkazy a skripty. SQL Developer samozřejmě podporuje i programování a ladění PL/SQL procedur, funkcí a balíčků.

Oracle SQL Developer také obsahuje rozšiřitelnou řadu připravených sestav umožňujících získávání informací o databázi. Oracle SQL Developer bude použit pro vývoj a správu datové vrstvy restauračního IS.



Obrázek 8. – Rozhraní Oracle SQL Developer

4 DATOVÁ ÚLOŽIŠTĚ A JEJICH ALTERNATIVY

Roli, kterou hraje databáze v každém softwarovém projektu, závisí hlavně na jeho typu. Některé aplikace databázi nepotřebují, jiné se však bez datového úložiště neobejdou. V některých případech slouží databáze jako privátní úložiště dat aplikace a není sdílena mezi více počítači. V těchto případech není kladen velký důraz na výkonnost, důležité je, aby databáze pracovala spolehlivě bez údržby a snadno se instalovala. V případě restauračního informačního systému musí systém pracovat s velkou, sdílenou databází, pro kterou tvoří zpracování dat podstatu jeho činnosti. Je tedy velmi důležitý správný návrh této databáze, jednak proto, že výkonnost představuje důležité kritérium, ale také z důvodu možnosti sdílení jedné databáze více aplikacemi. Musí být tedy přesně definována její struktura, odpovědnost za jednotlivé tabulky nebo celé segmenty, případně uživatelská práva. Je potřeba také stanovit pravidla pro provoz a zálohování.

V dnešní době existují dvě základní možnosti výběru databázového produktu.

První možnost je využít komerční databáze, které ovšem nejsou zadarmo a při jejím použití je potřeba se řídit přísnými licenčními podmínkami. Na druhou stranu však společnosti poskytující tyto produkty dávají stoprocentní technickou podporu, která se může stát nepostradatelnou v případě nasazení naší aplikace do komerčního prostředí.

Tyto komerční produkty však mají své hybridní zástupce tzv. express edice, kdy je jejich využití bezplatné, ale v určitých směrech je jejich použití omezeno.

Druhou možností je využít opensource databáze, kdy je její použití zdarma a v mnoha případech se poskytuje možnost získání zdrojového kódu daného produktu a tedy možnost jeho úpravu podle vlastních představ. Databázových technologií existuje celá řada, pro vývoj restauračního IS jsem se rozhodoval mezi Oracle 10g Express Edition a opensource databázovou technologií PostgreSql. Tyto dva produkty jsou si hodně podobné, avšak pro případné rozšíření datového úložiště a případnou možnost technické podpory byla zvolena Oracle 10g Express Edition.

4.1 Oracle 10g Express Edition

Oracle Database 10*g* Express Edition, je nejnižší edicí databázového serveru, který dominuje především v oblasti zpracování velkých objemů dat a podporou prostředí s vysokými počty transakcí a vysokými požadavky na zabezpečení dat. Express Edition sdílí s ostatními edicemi stejné jádro databázového serveru, včetně mechanismů řízení přístupu, které umožňují efektivně zvládat vysoké počty transakcí s minimem vzájemného blokování

4.1.1 Základní vlastnosti databázového jádra

. Základem je zamykání záznamů na úrovni řádků a tzv. Multi-Version Read Consistency, mechanismus, zajišťující že nedochází k vzájemnému blokování zápisových a čtecích operací. Také rozsah funkcí jazyka SQL není oproti vyšším verzím omezen. Je tedy například dostupná široká škála analytických funkcí nebo klauzule RETURNING umožňující vracet hodnoty zpět do prostředí aplikace přímo v běhu DML operací. Tato klauzule se uplatní například u generovaných primárních klíčů. Při natahování dat z jiných zdrojů může být zajímavá také možnost vkládat data pomocí příkazu INSERT podmínečně do více tabulek, nebo třeba operace MERGE pro slučování dat, která spojuje funkcionalitu UPDATE pro existující záznamy a INSERT pro záznamy neobsažené v cílové sadě.

Express Edition nabízí možnost implementace datové logiky v uložených procedurách a v široké škále triggerů vytvářených v osvědčeném jazyce PL/SQL. Tato edice databáze také obsahuje i širokou podporu práce s netradičními daty z pohledu relačních databází, jako jsou například XML dokumenty, textové dokumenty nebo prostorová data. V případě XML dat jde například o rozšíření jazyka SQL, jenž nese název SQL/XML, podporu XML Schémat, XML transformací atd. Textové dokumenty lze do databáze Oracle nejen ukládat, ale zároveň i vytvářet fulltextové indexy a provádět fulltextové vyhledávání přímo v rámci SQL. Podporovány jsou vedle prostých textů také dokumenty obvyklých formátů, jako je MS Office či PDF. Podpora prostorových dat v této edici dokonce vysoce přesahuje funkcionalitu většiny ostatních databázových systémů. [6]

4.1.2 Omezení 10g Express Edition

I když je Express Edition nejnižší edicí, a tudíž má oproti vyšším edicím některá omezení, může být i přesto výhodnou variantou pro menší aplikace. A to nejen jako prostředí vývojové, ale i provozní. Hlavními omezeními jsou 4 GB fyzické velikosti

datového úložiště, využitelnost pouze 1 GB RAM a 1 procesoru serveru. Pro menší firmy či menší aplikace nejsou tyto omezení limitujícím faktorem. Také na poměry Oracle omezená podpora platforem na Windows a Linux je pro tyto podmínky postačující.

4.2 PostgreSql

PostgreSql je plnohodnotným relačním databázovým systémem s otevřeným zdrojovým kódem. Běží na všech rozšířených operačních systémech včetně Linuxu, UNIXů a Windows. Splňuje podmínky ACID, plně podporuje cizí klíče, operace JOIN, pohledy, spouště a uložené procedury. Obsahuje většinu SQL92 a SQL99 datových typů,. K systému existuje volně dostupná dokumentace PostgreSQL je šířen pod BSD licencí, která je nejliberálnější ze všech opensource licencí. Tato licence umožňuje neomezené bezplatné používání, modifikaci a distribuci PostgreSQL a to ať pro komerční nebo nekomerční využití. PostgreSQL můžete šířit se zdrojovými kódy nebo bez nich, zdarma nebo komerčně. PostgreSQL umožňuje běh uložených procedur napsaných v několika programovacích jazycích, v Perlu, v Python, v jazyku C nebo v speciálním PL/SQL, jazyku vycházejícím z Oracle technologie. Předností systému PostgreSQL je rozšiřitelnost. Systém může být bezproblémově rozšiřován o nové datové typy, funkce operátory, agregační funkce, procedurální jazyky.

4.3 Přístup k persistentním datům

Při vývoji informačního systému pomocí Java technologií se nabízí základní dva přístupy k relační databázi. Lze použít standardní SQL přístup nebo libovolný dostupný ORM Framework, respektive Java Hibernate. SQL přístup využívá klasických DML a DDL příkazů a Java Hibernate využívá vlastní dotazovací jazyk HQL. Při vývoji restauračního IS bude použita alternativa SQL přístupu.

4.3.1 SQL přístup

Zkratka SQL znamená Structured Query Language, což je strukturovaný dotazovací jazyk. Jazyk v sobě zahrnuje nástroje pro tvorbu databází (tabulek) a dále nástroje na manipulaci s daty (vkládání dat, aktualizace, mazání a vyhledávání informací).

SQL patří mezi tzv. deklarativní programovací jazyky, což v praxi znamená, že kód jazyka SQL se nepíše v žádném samostatném programu (jako by tomu bylo např. u jazyka C++ nebo Java), ale vkládáme jej do jiného programovacího jazyka, který je již procedurální. SQL je založeno na relačním modelu. Zatímco relační model poskytuje teoretické základy, SQL je jazyk, jenž podporuje fyzickou implementaci databáze. Přestože je z důvodů jeho neprocedurální povahy za podjazyk, je kompletním jazykem, v němž je možné vytvářet a spravovat databázové objekty, zabezpečovat objekty a manipulovat s těmito daty. Příkazy SQL jsou rozděleny do podle funkce, kterou provádí:

- Jazyk definice dat (DDL – Data Definition Language)
- Jazyk pro řízení dat (DCL – Data Control Language)
- Jazyk pro manipulaci s daty (DML – Data Manipulation Language)

4.3.1.1 Jazyk definice dat

Příkazy DDL se používají k vytváření, úpravám nebo odstraňování databázových objektů, jako například tabulky, triggery atd. Klíčovými slovy SQL nejčastěji spojovanými s příkazy DDL jsou create, alter a drop. Například příkaz create table, lze použít k vytvoření tabulky v dané databázi. Příklad použití DDL je v příloze PII.

4.3.1.2 Jazyk pro řízení dat

Příkazy DCL umožňují řídit, kdo bude mít přístup k určitým objektům dané databáze. V jazyce DCL lze povolovat přístup nebo jej zamezovat pomocí příkazů grant nebo revoke k těmto objektům, jenž patří mezi primární příkazy DCL. Příkazy pro řízení dat také umožňují řídit typ přístupu každého uživatele k databázovým objektům. Lze tedy určit, kteří uživatelé budou moci určitou skupinu dat pouze prohlížet, a kteří budou moci manipulovat s daty.

4.3.1.3 Jazyk pro manipulaci s daty

Příkazy DML se používají k prohlížení, přidávání, úpravám nebo odstraňování dat uložených v databázových objektech. Primárními klíčovými příkazy jsou select, update, insert a delete. Tyto příkazy jsou používány nejčastěji. Například příkaz select lze použít k načítání dat z tabulky a příkaz insert pro přidání záznamu do tabulky.

4.3.2 Java Hibernate

Hibernate je výkonný objektově relační nástroj pro perzistenci objektů v aplikaci a správu databázových dotazů nad platformou jazyka Java. Hibernate umožňuje vývoj persistentních objektů v jazyce Java splňující základní vlastnosti, jako jsou asociace, dědičnost, polymorfismus, kompozice a práce s kolekcemi objektů v jazyce Java.

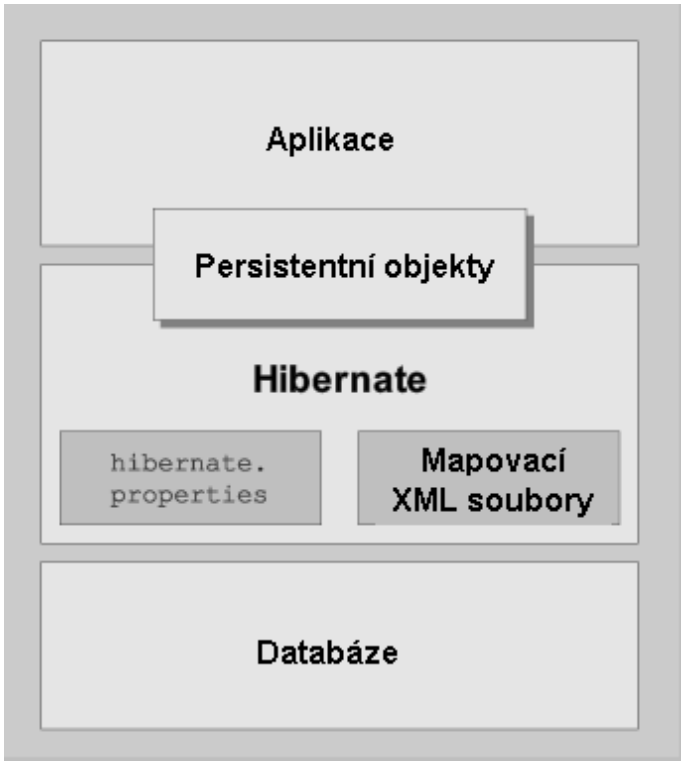
Hibernate odstraňuje generování kódu v době překladu systému nebo zpracování bitového kódu. Místo toho jsou použity principy reflexe a generování bitového kódu za běhu programu, kdy generování SQL dotazu se provádí až v době startu systému. Tento přístup zajišťuje, že Hibernate neovlivňuje překlad nebo inkrementální kompilaci.

Hibernate obsahuje i vlastní typový systém, který je mapován na specifické datové typy databázových strojů. Pro zajištění persistence základních tříd nemusí být implementována žádná zvláštní rozhraní ani nemusí být použity speciální mechanismy dědění z kořenových persistentních tříd. Hibernate také nepoužívá žádné zpracování během procesu překladu a sestavování aplikace, plně využívá mechanismu reflexe jazyka Java. To znamená, že základní třídy mohou být mapovány na jednotlivé databázové tabulky bez nutnosti uvádění jakýchkoliv vazeb ve zdrojových kódech daných tříd.

Jediným omezením kladeným na implementaci tříd určených pro persistenci dat, je přítomnost identifikačního atributu. Tento atribut slouží později k odlišení jednotlivých persistentních objektů na základě databázové identity. Hibernate podporuje tento požadavek sadou zabudovaných generátorů identifikátorů pro různé scénáře použití, od implicitních identifikátorů pro databázové sekvence až po implementace různých algoritmů pro generování identifikátorů.

Hibernate, jakožto objektově relační nástroj, je řízen konfiguračními soubory ve formátu XML. V těchto souborech jsou uvedena mapování aplikovaná na jednotlivé tabulky daného databázového schématu na základní třídy. Na základe těchto mapovacích

souborů je Hibernate schopen zajistit persistenci objektu. Mapovací soubory lze ale využít i při opačném postupu. Jsou-li nejdříve vytvořeny základní objekty a k nim odpovídající mapovací XML soubory, je možné z těchto souborů vygenerovat odpovídající databázová schémata. [7]



Obrázek 9. – Architektura Java Hibernate

4.3.2.1 HQL

Dotazovací jazyk Hibernatu (HQL - Hibernate Query Language) byl navržen jako minimální objektově orientované rozšíření základní obecné verze databázového dotazovacího jazyka SQL. HQL tím poskytuje elegantní přemostění mezi objekty a relačními architekturami databázových strojů. Přestože však HQL vychází ze syntaxe dotazovacího jazyka SQL, jedná se o plně objektově orientovaný jazyk, ve smyslu podpory dědění, polymorfismu nebo asociací.

5 GRAFICKÉ UŽIVATELSKÉ ROZHRANÍ

Grafické uživatelské rozhraní je rozhraní sloužící k interaktivní komunikaci mezi uživatelem a počítačem. Zpravidla každý novodobý počítačový program používá pro interakci grafiku. GUI je intuitivní a efektivní způsob jak shromažďovat informace od uživatele nebo mu je následně zobrazovat ke čtení. Základním stavebním kamenem GUI jsou standardní grafické prvky, které dohromady vytvářejí uživatelské rozhraní. Mezi tyto grafické prvky patří obvykle okna, nabídky, popisky, textová okna, přepínače a další podobné objekty rozhraní GUI, s nimiž se lze setkat téměř ve všech komerčních programech. Celá aplikace byla programována na Java platformě formou tlustého klienta. Nabízí se dvě základní knihovny pro tvorbu grafického uživatelského rozhraní.

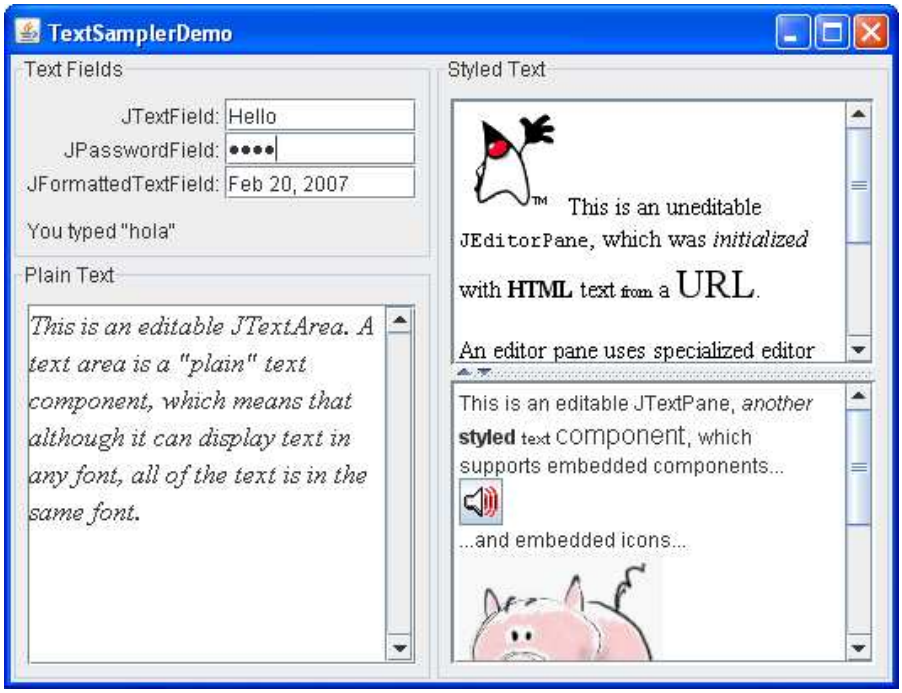
Java obsahuje více knihoven pro práci s GUI. Při vývoji aplikace byly použity Java Foundation Class (JFC) Swing, která je implementována v javax.swing balíčku a s ní Abstract Windows Toolkit (AWT), který JFC doplňuje.

Vytvoření grafického uživatelského rozhraní v Javě vyžaduje čtyři základní prvky:

- Komponenty
- Správce rozvržení
- Kontejner
- Obsluha událostí

5.1 Komponenty

V Javě je každá položka grafického uživatelského rozhraní (popisky, okna, přepínače a další) komponentou. Komponenty jsou hlavními grafickými prvky, se kterými uživatel přímo pracuje. Bez ohledu na jejich konkrétní funkce mají všechny komponenty společnou sadu metod a specifických vlastností určenou k nastavení barev, velikosti a podobně. Každá komponenta má dále metody určené přímo ke své funkci. Tyto komponenty lze najít v balíku javax.swing.



Obrázek 10. – Příklad některých swing komponent

5.2 Kontejner

Všechny komponenty GUI musí být uspořádány do kontejneru. Java obsahuje několik typů kontejnerů. Na kontejner se lze dívat jako na virtuální krabici, do které se vkládají prvky grafického uživatelského rozhraní. Nejběžnějším kontejnerem je JFrame, který představuje základní okno celé aplikace. Tento kontejner je složitější, protože obsahuje řadu metod a vlastností a příslušné programové omezení. Jednodušším, ale také často používaným kontejnerem je kontejner JPanel, který představuje panelový objekt. Tyto a další méně používané kontejnery lze použít z balíku javax.swing a java.awt.

5.3 Správce rozvržení

Při přidání komponenty GUI do kontejneru správce rozvržení určí správce rozvržení, kam bude komponenta v rámci celého kontejneru umístěna. Java poskytuje šest standardních správců rozvržení, přičemž každý z nich rozmístí komponenty různým způsobem. Správce rozvržení je automaticky spojován s každým kontejnerem při jeho vytváření, avšak podle potřeby lze explicitně správce rozvržení libovolně měnit. Všechny správce rozvržení lze najít v obou balících java.awt a javax.swing.

5.4 Obsluha událostí

Obsluha událostí je nejsofistikovanějším prvkem při tvorbě grafického uživatelského rozhraní. Událost je požadovaná akce, která se provede například po klepnutí myši na příslušné aktivní tlačítko nebo vstupu z klávesnice. Klepnutím na tlačítko myši nebo stisknutím klávesy se vytvoří událost, která je objektem Javy. Všechny události jsou obsluhovány vytvořením třídy naslouchání, které sledují určitý typ události a provedou patřičnou akci (metodu), která se nazývá obsluha události. Třídy naslouchání implementují rozhraní naslouchání, která specifikují jména metod obsluhy událostí určených k ošetření určité události. Rozhraní naslouchání lze najít v balíku java.awt.event.

II. PRAKTICKÁ ČÁST

6 INFORMAČNÍ SYSTÉM PRO RESTAURAČNÍ ZAŘÍZENÍ

Aplikace byla navržena a naprogramována pro Java platformu s využitím databáze Oracle 10g Express Edition. Důraz byl kladen především na pochopitelné a jednoduché ovládání celého systému a jeho další modifikaci a případná rozšíření.

6.1 Hardwarové a softwarové požadavky

U každého informačního systému hraje velmi důležitou roli jeho správný a bezproblémový chod a v neposlední řadě i jeho výkonnost. Výkonnost aplikace může významně ovlivnit celkový návrh aplikace, architektura a hardware, jež informačnímu systému poskytují prostředky pro svou funkčnost. Architektura systému je dvouvrstvového charakteru. Obsahuje klientskou a datovou vrstvu, proto je nutné stanovit minimální hardwarové a softwarové požadavky pro každou vrstvu zvlášť.

6.1.1 Hardwarové požadavky

Klientská vrstva musí splňovat následující minimální požadavky:

- Procesor: 1GHz Intel Pentium III nebo AMD Athlon nebo ekvivalentní
- RAM: 256 MB
- Volné místo na pevném disku: 100 MB

Datová vrstva (datový server) musí splnit minimální nároky jako vrstva klientská, avšak HW nároky jsou omezeny použitou databází:

- pouze 1 procesor datového severu
- 4 GB fyzické velikosti datového úložiště
- operační paměť max. 1 GB RAM

6.1.2 Softwarové požadavky

Řešený informační systém má dva hlavní softwarové požadavky. Prvním požadavkem je operační systém Microsoft Windows XP Home a vyšší nebo jakákoliv dostupná linuxová distribuce. Základním stavebním kamenem celého restauračního systému je Java platforma, proto musí mít cílová stanice nainstalovanou Java Virtual Machine. Tento požadavek lze interpretovat jako omezující faktor, avšak celá aplikace běží právě na JVM, a to z důvodu multiplatformosti celého informačního systému. Konfigurace virtuálního stroje není potřeba, aplikace běží s implicitním nastavením jenž je nastaveno přímo po instalaci JVM.

6.2 Spouštění aplikace a ukončení aplikace

Celá aplikace je uložena v jediném souboru, který je nazván *diplomka.jar*. Tento soubor lze umístit libovolně v celé adresářové struktuře operačního systému. Pro větší efektivnost byl vytvořen zástupce aplikace na pracovní ploše operačního systému. Aplikace se spustí kliknutím na zástupce. Aplikace se ukončí kliknutím na křížek v pravém horním rohu hlavního okna. Při zavření okna se provede automatické odhlášení uživatele ze systému.

7 MODULY IS PRO RESTAURAČNÍ ZAŘÍZENÍ

Informační systém byl navržen a realizován ve formě jednotlivých funkčních částí, tzv. modulů. Každý z těchto modulů obsahuje příslušnou aplikační logiku a způsob vykreslení grafického uživatelského rozhraní. Aplikační modul je realizován sadou Java tříd a je importován do formy Java balíčku. Mezi jednotlivými třídami, respektive moduly dochází k vzájemné interakci z důvodu správné funkčnosti aplikace, při dodržení podmínek volné vazby a silné soudružnosti dané třídy. Všechny třídy byly realizovány tak, aby pokud možno splňovali všechny aspekty objektově orientovaného programování.

Struktura Java balíčků restauračního IS:

- *diplomka.api* – obsahuje třídy pro aplikační logiku systému
- *diplomka.db* - obsahuje třídy pro interakci s databází
- *diplomka.gui* – třídy pro vykreslení grafického uživatelského rozhraní
- *diplomka.print* – třída realizující tisk objednávky
- *diplomka.util* – podpůrné třídy

Jednotlivé moduly (submoduly) restauračního IS:

- Autentifikační modul
- Modul hlavní nabídky
- Jídelníček
- Nastavení
- Restaurace
- Bar
- Platba objednávky
- Tisk objednávky

7.1 Struktura hlavního okna

Hlavní okno se skládá z několika oddělených panelů, jež jsou zobrazeny po celou dobu chodu aplikace. Záhlaví celého systému tvoří grafická lišta, která nemá žádnou funkčnost, jedná se tedy o vizuální prvek aplikace. Pod touto lištou je umístěn informační panel se základními údaji, kterými jsou jméno restaurace využívající řešený systém, jméno přihlášeného uživatele a aktuální datum a čas. Další panel, který se nachází pod záhlavím, je panel s tlačítky pro práci s některými moduly restauračního informačního systému, například hlavní menu, bar, jídelníček, restaurace a nastavení. Pro přehlednost a snadnou orientaci v systému byl přidán grafický prvek, jenž způsobí, že tlačítko představující aktuálně spuštěný (aktivní modul) bude mít odlišnou barvu než tlačítka zastupující ostatní moduly. Uprostřed se nachází hlavní panel, jenž je dynamický a zobrazuje grafické uživatelské rozhraní právě používaného modulu. Zápatí systému zobrazuje informační lištu o výrobci systému. Další funkčností, kterou nabízí hlavní okno, je, pokud to aplikační logika daného procesu vyžaduje, popup (vyskakovací) okno, obsahující zpravidla další doplňující funkčnost při práci se systémem. Pokud uživatel toto okno aktivoval omylem, nabízí se možnost jej zavřít pomocí funkčního tlačítka „Cancel“, jinak se okno zavře v závislosti na dané situaci s provedením příslušné aplikační logiky. V mnoha komerčních informačních systémech lze otevírat popup okna donekonečna, což obvykle snižuje snadnou orientaci v systému. Aby se zamezilo této negativní orientační vlastnosti, stanovuje systém omezení pouze na jedno popup okno. Logika systému je navržena tak, aby uživatel nepotřeboval v jednom čase použít více než jedno popup okno.

Další důležitou vizuální vlastností je velikost okna. Velikost okna je implicitně nastavena přes celou výšku a šířku obrazovky monitoru. Parametry okna se vypočítají podle aktuálního rozlišení monitoru. Nemůže se tak stát, že okno bude zpočátku zmenšeno, což může také ovlivnit efektivitu při práci se systémem. Avšak uživatel může měnit velikost okna libovolně a přitom nedojde k překrývání komponent, ani dalšímu nepříznivému omezení v grafickém uživatelském rozhraní. Aby systém splnil tuto podmínku, mají všechny komponenty v GUI relativní rozměry. Vývoj komponent s relativními rozměry je složitější, ale výsledné GUI je dynamičtější.

7.2 Modul autentifikace (autentifikace uživatele)

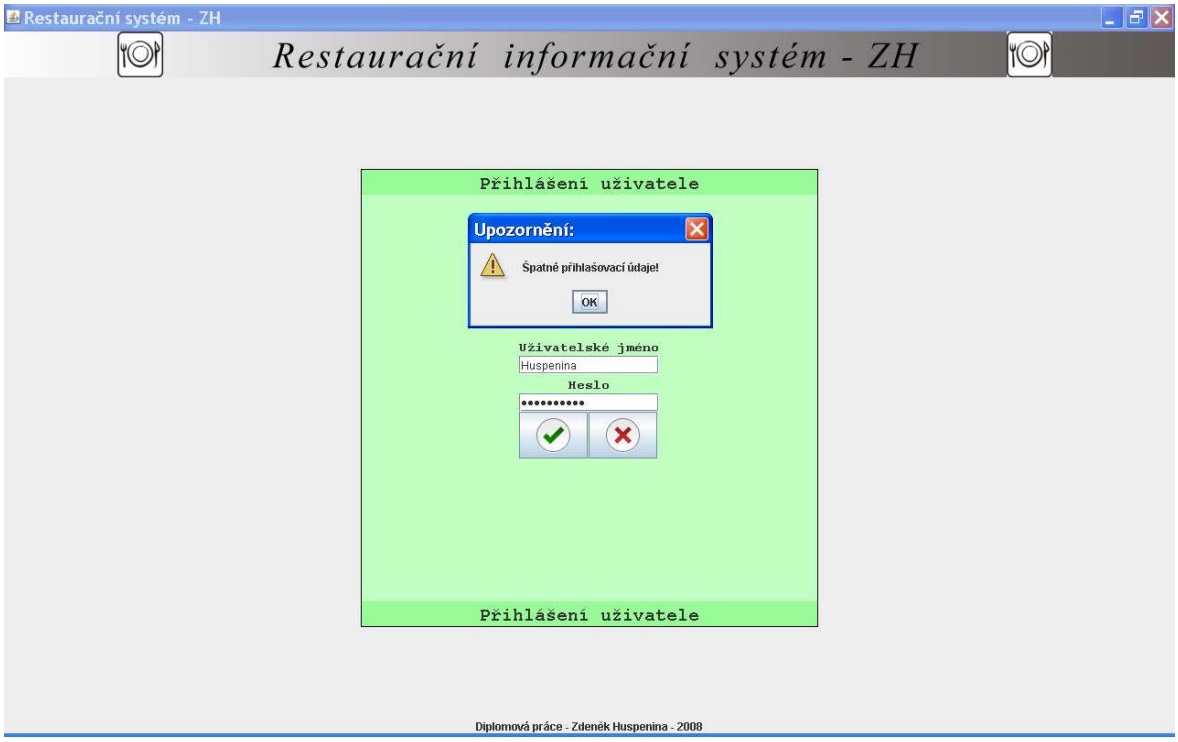
Jedním z hlavních požadavků na řešený restaurační informační systém je, aby byl pro více uživatelů. Proto je nejdříve potřeba každého unikátního uživatele při přihlášení do systému autentifikovat a ověřit tak jeho přihlašovací údaje. Autentifikačními údaji jsou libovolné uživatelské jméno a heslo. Při přihlašování verifikuje systém tyto údaje, to znamená, pokud existuje uživatelské jméno s příslušným heslem, je uživatel přihlášen do systému a může začít pracovat, přičemž se otevře hlavní nabídka pro práci se systémem. Jestliže správná dvojice uživatelské jméno-heslo neexistuje, objeví se potencionálnímu uživateli upozornění o nesprávnosti zadaných údajů.

Protože systém obsahuje i část pro modifikaci jednotlivých záznamů v databázi, jenž nemůže upravovat každý uživatel stejně tak i prohlížení všech uzavřených objednávek, je zapotřebí rozdělit systém do veřejných a privátních sekcí. Veřejné sekce jsou určeny pro všechny uživatele bez ohledu na oprávnění vstupu, mezi které patří například přístup do modulu Bar, a to z důvodu, že tento modul obsahuje celý proces a aplikační logiku pro práci s objednávkami a zákaz přístupu do této části by tak byl zcela iracionální. Při vytvoření uživatele je potřeba určit, jaká bude mít daný uživatel oprávnění pro vstup do privátních částí systému. Tato oprávnění určuje zpravidla management daného podniku nebo zodpovědný provozní manager, který se tak stává administrátorem celého informačního systému.

Jestliže je daná sekce uživateli zakázána, je tlačítko neaktivní (šedá barva) a není možno do sekce vstoupit. Privátní sekce je tedy zakázána na nejvyšší možné úrovni a neoprávněnému uživatel je tedy zakázán i náhled do této sekce.

Restaurační informační systém má následující privátní sekce:

- Vytvoření a modifikace stálých zákazníků a jejich slev
- Vytvoření a modifikace uživatelů restauračního systému
- Vytvoření a modifikace stolů v restauraci
- Prohlížení všech uzavřených objednávek

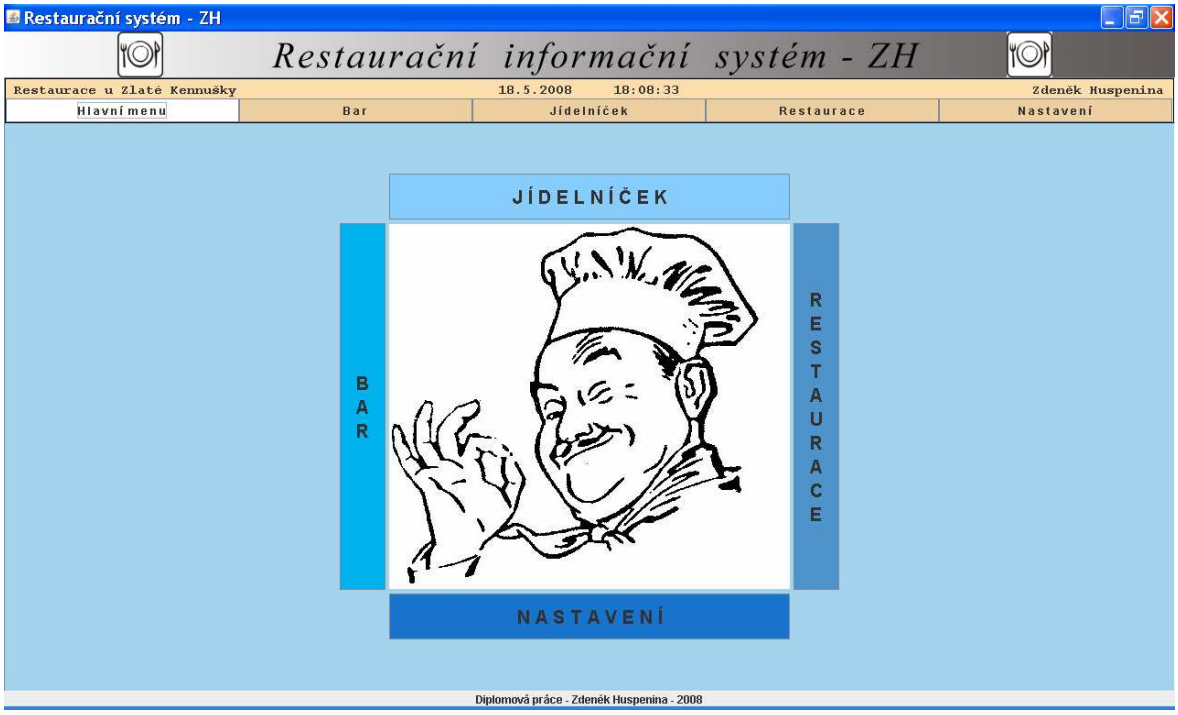


Obrázek 11. – Autentifikace uživatele do systému

Okno autentifikačního modulu obsahuje dvě vstupní textová pole pro zadání uživatelského jména a hesla. Z bezpečnostních důvodů se text při psaní uživatelského hesla nezobrazuje, zobrazují se pouze černé znaky, proto lze při potencionálním útoku zjistit pouze počet znaků hesla. Kliknutím na zelené tlačítko se provede samotná verifikace, kliknutím na červený křížek dojde k ukončení práce se systémem.

7.3 Modul hlavní nabídky

Modul hlavní nabídky se zobrazí automaticky po úspěšném přihlášení uživatele do systému. Jedná se o úvodní grafický rozcestník, který slouží zpravidla pro rychlou orientaci při prvotní fázi práce se systémem. Dává uživateli možnost výběru práce s příslušným modulem. Aktivace modulu se provede kliknutím na některé z funkčních tlačítek. Každé funkční tlačítko je obohaceno o grafické prvky, například po přejetí kurzorem se vytvoří po obvodu tlačítka rámeček s odlišnou barvou.



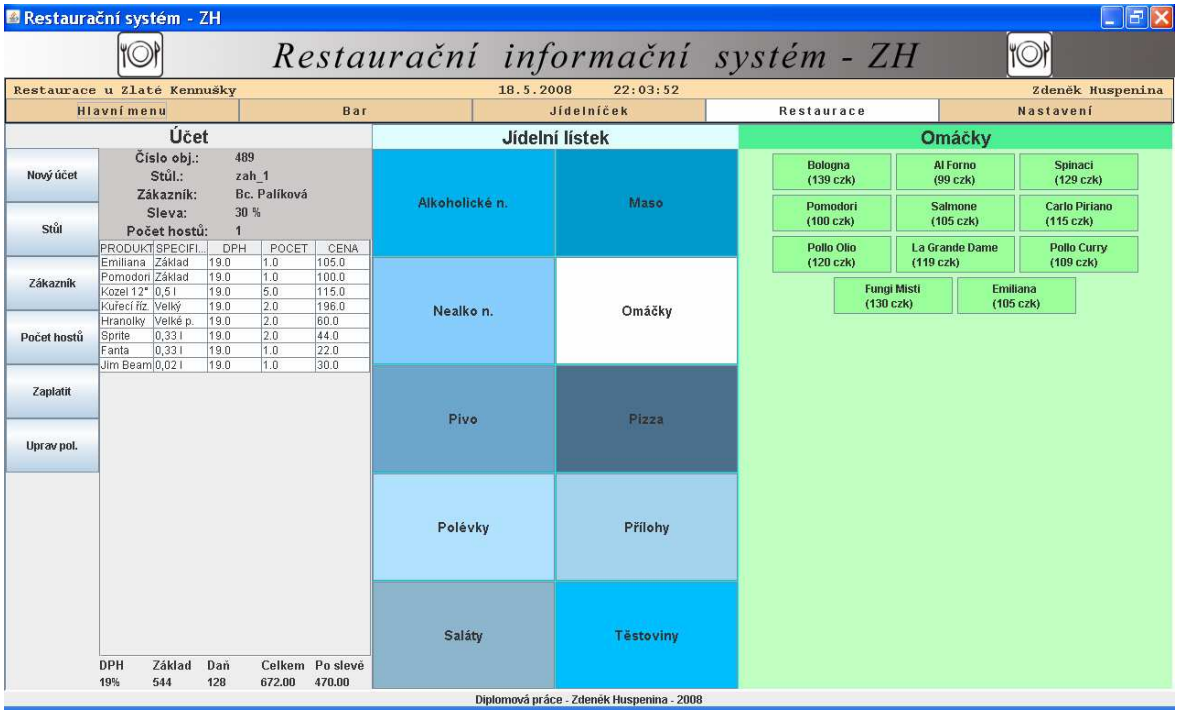
Obrázek 12. – Modul hlavní nabídky

7.4 Modul Bar

Modul Bar je nejdůležitějším modulem celého systému. Obsahuje složité grafické uživatelské rozhraní a důležitou aplikační logiku pro vytváření a modifikaci restauračních objednávek. Z důvodu velké složitosti byl tento modul strukturován do několika jednodušších submodulů, jako například platba objednávky a další. Každý z těchto submodulů je zodpovědný za realizaci příslušné části objednávky. Mezi jednotlivými submodely dochází k interakci a tím tvoří ucelený celek - modul baru. Modul Bar se aktivuje kliknutím na tlačítko Bar v nabídce hlavního menu nebo v tlačítkové liště umístěné pod záhlavím hlavního okna. Modul lze aktivovat kliknutím na příslušnou objednávku v nabídce „Restaurace“, pokud není objednávka uzavřena, aktivuje se modul a otevře se daná objednávka. Lze tak pořizovat další položky objednávky nebo modifikovat informace o objednávce.

Grafické uživatelské rozhraní bylo rozděleno do dvou vertikálních panelů. Levá část zobrazuje entitu objednávky s příslušnými informacemi o dané objednávce, jako jsou například číslo objednávky, umístění vztahující se k fyzické struktuře rozložení stolů po restauraci. Dále panel zobrazuje funkční tlačítka pro modifikaci objednávky, modul placení, možnost založení nové objednávky a další funkční možnosti. Pravý panel slouží

jako nabídka produktů restauračního zařízení, které je možno importovat do objednávky jako její položky.



Obrázek 13. – Modul Bar

Jednotlivé submodely modulu Bar:

- Submodul Účet
- Submodul Nabídka produktů (Jídelníček)
- Submodul Platba účtu
- Submodul Tisk účtu

Každý z těchto submodulů má specifické vlastnosti a funkčnost, přičemž budou popsány níže.

7.4.1 Submodul Účet

Submodul Účet slouží obecně pro práci s restauračními objednávkami. Obsahuje aplikační logiku a funkční tlačítka pro vytvoření nové objednávky, modifikaci již existujících objednávek a jejich položek a další. Grafické uživatelské rozhraní bylo rozděleno do několika částí. Samotný účet, jenž představuje druhou část, se skládá ze tří částí, jimiž jsou hlavička účtu, obsah účtu a sumační lišta. Hlavička obsahuje základní informace o daném účtu, jeho umístění, a jedná-li se o stálého zákazníka, tak jeho přiřazenou slevu a údaj o počtu hostů. Tento submodul patří do modulů veřejných, proto je k dispozici všem uživatelům bez ohledu na jejich oprávnění.

7.4.1.1 Slevový systém

Zpravidla každý objednávkový informační systém zaměřený na objednávku různého typu produktů má implementován systém slev. Řešený restaurační IS tuto možnost taky nabízí. Slevový systém vychází z evidence stálých hostů, VIP a firemních zákazníků nebo zaměstnanců, přičemž každý má příslušné cenové zvýhodnění a individuální cenovou politiku. Případné využití slevového systému zvyšuje prestiž a jméno restauračního zařízení využívající IS. Protože se jedná o zásadní ovlivnění cenové politiky restaurace, má možnost pořizovat a modifikovat stálé zákazníky pouze oprávněná osoba.

Tato osoba zapíše jméno zákazníka a jeho slevu do systému. Slevový interval se pohybuje v rozmezí od 0 do 100 procent. Jestliže při vytvoření objednávky nebude vyplněna informace o stálém zákazníkovi, bude systém pracovat s hodnotou 0 procent a účtenka nebude mít žádnou slevu. Celková koncová cena na účtu, kterou musí zákazník zaplatit, je počítána jako cena s DPH násobená podílem slevy a konstantní hodnoty 100.

7.4.1.2 Vytvoření nového účtu

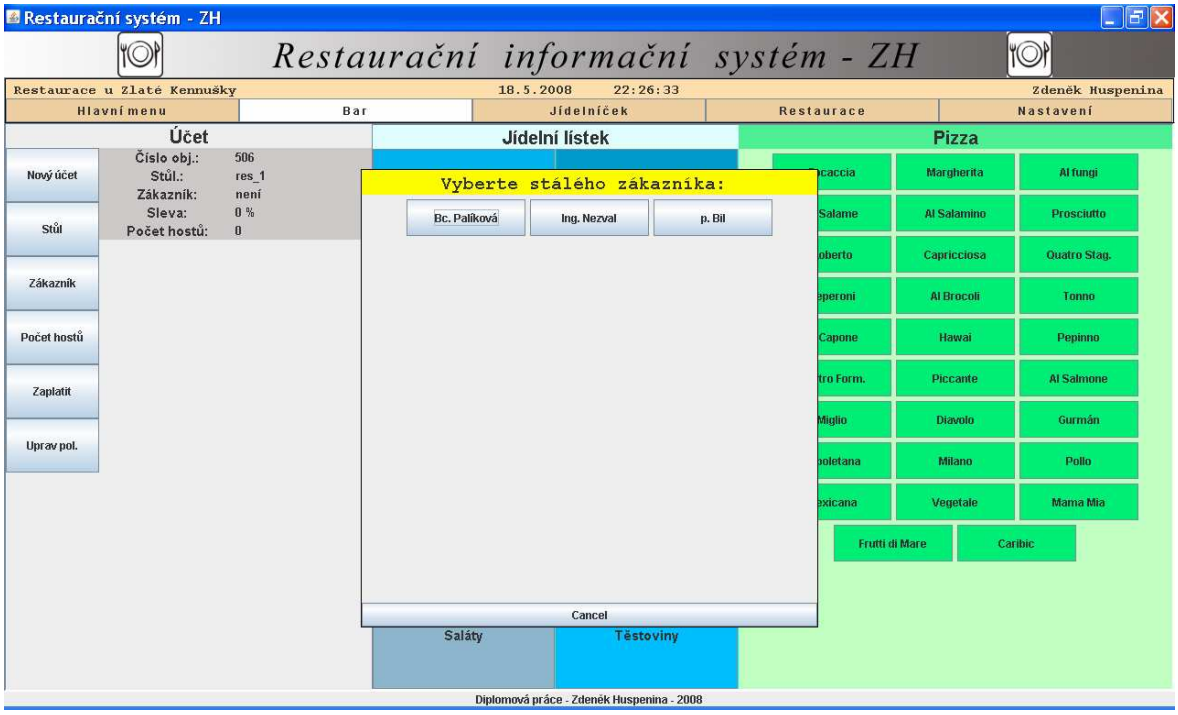
Základní funkcí v restauračním informačním systému musí být možnost vytvoření nové objednávky. Každá objednávka má specifické parametry, které je zapotřebí vyplnit. Parametry účtu (objednávky):

- Číslo objednávky
- Stůl (umístění obj.)
- Zákazník

- Sleva
- Počet hostů

Jestliže chce uživatel zahájit proces vytváření objednávky, musí aktivovat funkční tlačítko „Nový účet“. Aktivace tohoto tlačítka má za následek vygenerování nového čísla objednávky a zápis čísla hlavičky do objednávky. Každá objednávka má unikátní číslo, proto nikdy nenastane případná duplicita. Unikátnost se technicky realizuje pomocí inkrementálního klíče, respektive pomocí SQL sekvence a použitím triggeru nad příslušnou relační tabulkou. Nová objednávka je tedy založena a je potřeba doplnit její další údaje. Při vytvoření objednávky se automaticky vyplní aktuální datum a čas. Tato informace je velmi důležitá pro další zpracování objednávky, jedná se však o interní údaj, proto není vyplněn v hlavičce. Management restauračního zařízení má možnost nahlédnout do všech otevřených i uzavřených objednávek a v časové závislosti je pak třídit.

Po vygenerování čísla objednávky a zápisu data a času se zobrazí popup okno s možností výběru místa v restauraci. Možnosti výběru umístění jsou dynamicky vypisovány z databáze, kterou obhospodařuje zpravidla provozní restaurace nebo management. Každý účet se musí proto vztahovat k příslušnému místu objednávky (stolu), z toho důvodu je tato informace nepostradatelná. Pokud bude uživatel chtít přeskočit možnost výběru umístění objednávky, systém zobrazí varovnou hlášku o nesprávném postupu a uživatel bude požádán o nápravu, uživateli tedy nezbývá nic jiného, než umístění účtu vyplnit. Umístění se poté zobrazí v hlavičce. Dalším parametrem pro vyplnění je zákazník. Uživatel si vybírá z množiny předdefinovaných zákazníků s příslušnou slevou. Slevový systém bude popsán níže. Jestliže zákazník restauračního zařízení není stálý host, uživatel klikne na tlačítko „Cancel“ a systém doplní do hlavičky informaci o neznámém zákazníkovi a vyplní slevu 0 %. V dalším kroku se zobrazí vstupní číselník s nepovinnou informací o počtu hostů u daného stolu, respektive účtu. Implicitně je nastavena hodnota 1 (1 host), avšak tuto hodnotu lze měnit bez jakéhokoliv omezení. V této chvíli jsou vyplněny všechny povinné i nepovinné informace o daném účtu. Uživatel nyní může zahájit proces vyplňování položek příslušné objednávky (viz. kapitola 7.4.2).



Obrázek 14. – Vytvoření nového účtu – volba zákazníka

7.4.1.3 Modifikace hlavičky účtu

Jestliže není objednávka uzavřena, má možnost obsluha systému upravit údaje objednávky. Možnost modifikace se nabízí z důvodu špatného vyplnění informací a tedy jejich následnou opravu. Veškeré změny se realizují formou funkčních tlačítek umístěných v levé části panelu Účet. Dalším argumentem pro změnu údajů je změna umístění, to znamená, že se zákazníci přesunou od jednoho stolu k druhému. Tato situace je v praxi velmi obvyklá, uživatel systému tak může změnit umístění objednávky. Další možností změny na objednávce je zákazník. Tato situace v praxi nastane ojediněle, například když se ke stolu přidá zákazník, jenž má vyšší slevu než přítomní zákazníci restaurace, z toho důvodu musí být systém schopen takový požadavek splnit.

Informace s možností modifikace:

- Stůl – možnost výběru předdefinovaných stolů v popup okně, jestliže uživatel klikne na volbu „Cancel“, systém ponechá původní umístění objednávky.
- Zákazník - možnost výběru předdefinovaných stolů v popup okně, „Cancel“ ponechá původního zákazníka. Jestliže uživatel vybere jiného stálého

zákazníka s odlišnou slevou jako původní zákazník, systém samozřejmě přepočítá položky objednávky a ceny objednávky.

- Počet hostů – možnost výběru z číselníku, údaj informativního charakteru.

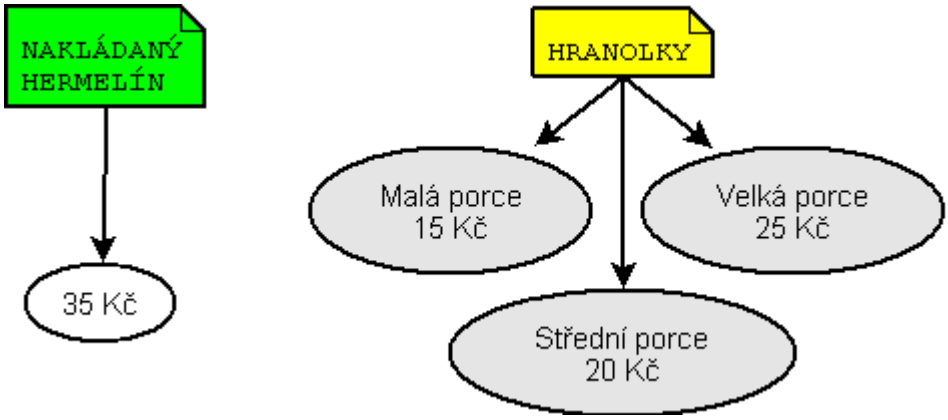
7.4.2 Submodul Nabídka produktů (Jídelníček)

Tento submodul realizuje možnost pořízení položek příslušné objednávky a jejich případnou modifikaci. Jedná se o množinu nabízených produktů restauračního podniku. Úzce spolupracuje se submodulem Účet, proto mezi nimi dochází k častým a četným interakcím. Spolu pak tvoří ucelený celek pro pořizování objednávek a jejich položek. Submodul Nabídka produktů (Jídelníček) obsahuje grafické uživatelské rozhraní a aplikační logiku pro pořizování položek. GUI bylo rozděleno do dvou vertikálních panelů.

Levý panel obsahuje hlavní sekce jídelníčku, například nealkoholické nápoje, pizza, přílohy atd. Sekce jídelníčku jsou údaje generované z databáze. Každá tato sekce zpravidla obsahuje podmnožinu produktů. Tuto podmnožinu zobrazuje v GUI pravý panel. Kliknutím na libovolnou sekci jídelníčku (levý panel) se zobrazí jeho položky v pravém panelu. Funkční tlačítko právě aktivované sekce bude mít bíle pozadí, jedná se o grafický prvek, jenž slouží k snazší orientaci v submodulu. Uživatel si tak vybírá příslušné položky k objednávce podle logického uspořádání.

7.4.2.1 Vytvoření nové položky na účtu

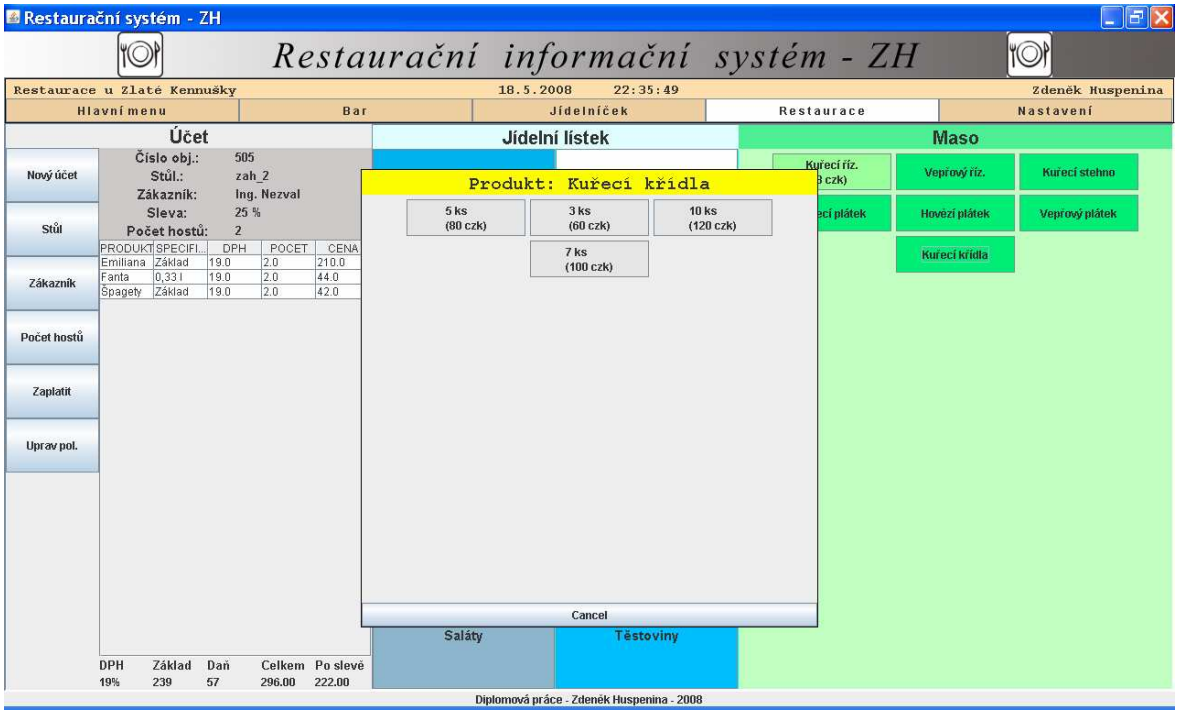
Vytvoření nové položky na účtu příslušné objednávky je velmi jednoduché. Uživatel si nejdříve zvolí sekci jídelníčku, ze které chce daný produkt vybrat. Otevře se mu tedy nabídka možných produktů v dané sekci s příslušným názvem a cenou za produkt na dalším řádku. Systém rozlišuje základní 2 druhy nabízených produktů, dle jejich další detailní specifikace. Mezi tyto informace patří cena za produkt, název, měrná jednotka, počet a daň z přidané hodnoty. Specifikace produktu se realizuje v modulu Jídelníček a při vytváření nové položky je specifikace již připravena.



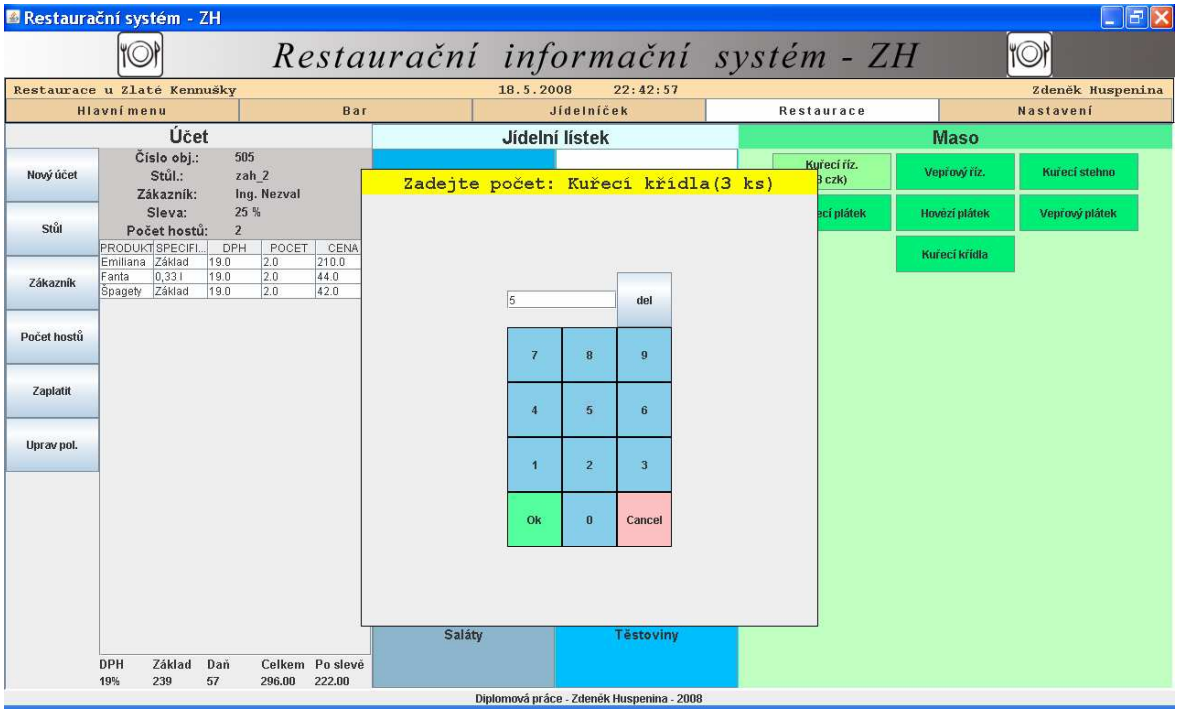
Obrázek 15. – Produkt prvního a druhého druhu

První druh produktu obsahuje pouze jednu sadu detailních informací a funkční tlačítko má vždy světle zelenou barvu. Produktem prvního druhu může být například nakládaný hermelín, jenž má pouze jednu specifikaci a nemá další složitější dělení. Kliknutím na funkční tlačítko prvního druhu má za následek aktivaci popup okna s číselníkem. Tímto číselníkem je uživatel požádán o zadání vstupu počtu položek daného produktu. Implicitně je hodnota v číselníku nastavena na 1 (počet daného produktu je 1). Kliknutím na „Cancel“ se zruší vytváření nové položky. Kliknutím na „Ok“ se provede import položky do objednávky a systém vypočte základ bez DPH, cenu jenž tvoří DPH, cenu celkem s DPH a koncovou cenu po případné slevě. Všechny tyto informace se zobrazí v sumační liště účtu. Položka objednávky se zobrazí ve středovém panelu účtu. Jestliže již položka na objednávce existuje, systém nepřipíše nový řádek, ale automaticky modifikuje příslušný řádek, tzn., aktualizuje sloupce počet a cena.

Druhý druh obsahuje vždy více než jednu sadu detailních informací. Mezi tento případ lze zařadit například desetistupňové pivo, jež může být malé, velké atd. Každá z těchto užších specifikací má tudíž svou jednotnou cenu, proto je potřeba rozlišovat detailní specifikace. Funkční tlačítko druhého druhu má tmavě zelenou barvu. Kliknutím na toto tlačítko se aktivuje popup okno s výpisem bližší specifikace daného produktu. Produktem druhého druhu mohou být například hranolky, které jsou rozděleny do třech specifikací, podle ceny a množství na malé, střední a velké. Uživatel si tedy vybere bližší specifikaci produktu. Dalším krokem je volba počtu položek daného produktu a pak následný import položek do objednávky a výpočet sumační lišty.



Obrázek 16. – Vytvoření nové položky produktu druhého druhu



Obrázek 17. – Vytvoření nové položky – počet položek produktu

7.4.2.2 Modifikace položek účtu

Úprava je umožněna z důvodu nevhodného pořízení položek účtu, jež je způsobeno obsluhou systému. Modifikace je povolena pouze u otevřených objednávek a aktivuje se funkčním tlačítkem „Uprav pol.“. Kliknutím na toto tlačítko se otevře popup okno, které je rozděleno na dvě horizontální části (panely). Horní panel obsahuje seznam všech položek aktuálního účtu. Kliknutí na sloupec „Č.p.“, jenž charakterizuje číslo položky otevře informace o položce ve spodním panelu. Uživatel může měnit pouze atribut počet. Jestliže uživatel nastaví počet na hodnotu 0, systém automaticky vymaže celou položku. Aktuální položku lze také odebrat pomocí funkčního tlačítka „Vymazat“ v nabídce modifikačního okna. Každá změna položky na objednávce se neprodleně projeví na výstupu objednávky, tedy sumační liště.

7.4.3 Submodul platba účtu

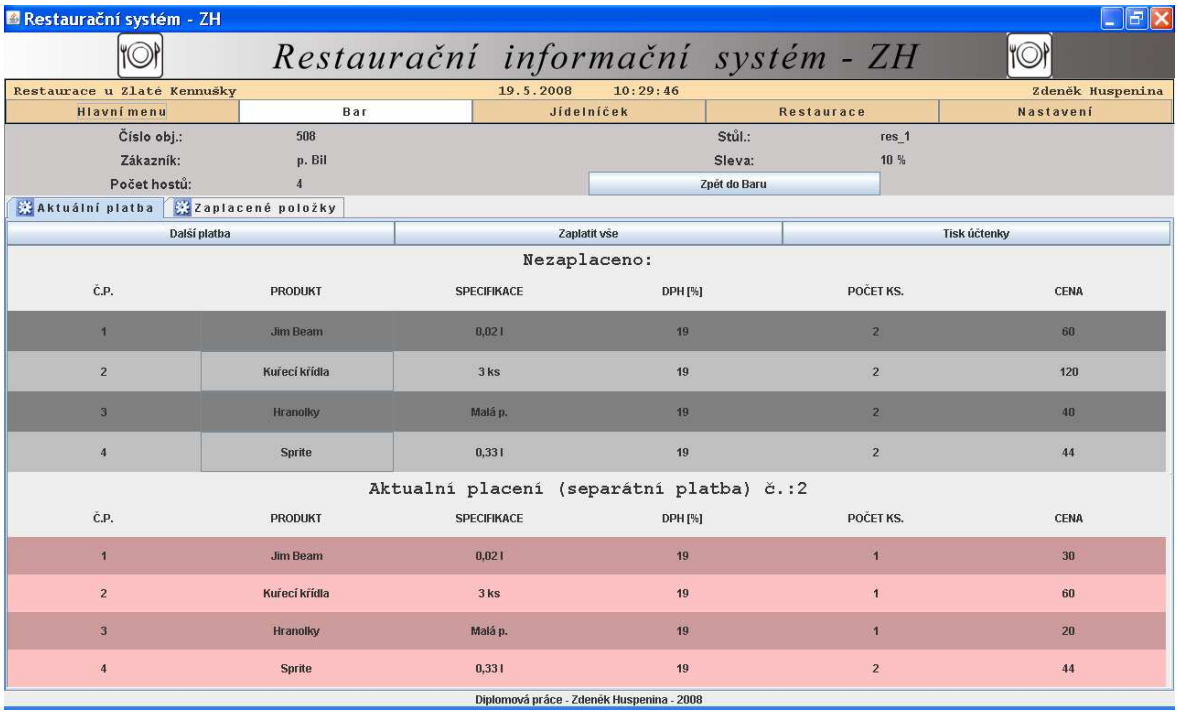
Platba účtu je proces, kdy zákazník restauračního zařízení nemá již žádné další požadavky na obsluhu a chystá se zpravidla opustit dané zařízení, přičemž je jeho povinností zaplatit za objednané produkty. V běžné praxi je více zákazníků přiřazeno k jednomu shodnému účtu, proto je aplikační logika řešeného informačního systému přizpůsobena možnosti separátní platby položek. Grafické uživatelské rozhraní bylo rozvrženo na několik panelů. Panel, který je umístěn pod hlavní nabídkou celého systému, obsahuje informativní údaje o aktuálně otevřené objednávce. Dále obsahuje tlačítko „Zpět do Baru“, jenž zajistí přesun zpět do modulu Bar, přičemž bude stále otevřena aktuální objednávka, uživatel tak má možnost pokračovat v práci s objednávkou a jejími položkami.

Struktura submodelu je dále rozdělena na dvě záložky a každá obsahuje vlastní aplikační logiku. Těmito záložkami jsou aktuální platba a zaplacené položky.

Záložka aktuální platba obsahuje funkční tlačítka pro práci se submodulem:

- Další platba – možnost další separátní platby na příslušné objednávce. Při volbě další separátní platby se každá tato platba identifikuje pomocí unikátního čísla, jenž je generováno inkrementálním klíčem.
- Zaplatit vše – funkční tlačítko pro zaplacení všech nezaplacených položek účtu najednou.
- Tisk účtenky – vytiskne samostatnou separátní nebo celkovou účtenku.

Tato záložka obsahuje dva dynamické seznamy, které specifikují seznam nezaplacených a aktuálně zaplacených položek dané objednávky. První seznam tedy vypisuje všechny nezaplacené položky. Spodní seznamový panel vypisuje položky, jež patří k aktuální separátní objednávce.



Obrázek 18. – Submodul platba účtu

Proces platby začíná kliknutím na položku ve sloupci „produkt“ v seznamu doposud nezaplacených objednávek. Následkem toho se aktivuje popup okno s číselníkem. V tomto případě slouží číselník jako vstupní pole pro zadání požadovaného počtu dané položky (produktu) k zaplacení. Implicitně se nastaví maximální možný počet, jenž se vyskytuje na položce, například je-li v položce objednávky 20 minerálních vod stejného druhu, nastaví se počet k zaplacení na hodnotu 20. Tuto hodnotu může uživatel libovolně měnit. Jestliže však nastaví větší počet, než je počet maximální, bude upozorněn ve formě varovného okna s možností změny hodnoty na maximální možný počet. Zvolený počet a tedy i cena se odpočítá z celkové objednávky a připočítá se do aktuální separátní objednávky. Tento proces je tedy transakčního charakteru. Jestliže uživatel již zaplatil všechny své položky, obsluha systému vytiskne objednávku aktivací funkčního tlačítka „Tisk účtenky“. Obsluha tak může pokračovat v objednávání nebo případném dalším procesu platby. V případě, že jsou již zaplaceny všechny položky objednávky, systém objednávku uzavře a nelze ji již

dále používat. Uživatel je o tomto faktu upozorněn v oznamovacím okně. Systém automaticky zapíše datum a čas uzavření objednávky, potom se přesune opět do základního modulu Bar. Pouze oprávněná osoba může objednávku dále jen prohlížet. Záložka „Zaplacené položky“ zobrazuje seznam, již zaplacených položek daných objednávek.

7.4.4 Submodul tisk účtu

Výstupem z restauračního informačního systému je vytisknutá účtenka. Slouží tedy jako doklad o zaplacení objednaného zboží zákazníkem. Tisknout lze oba druhy objednávek, a to separátní i celkovou objednávku. Tisk separátního účtu se provede v submodulu platba účtu aktivací funkčního tlačítka „Tisk účtenky“. Tisk celkové objednávky se provede automaticky, jsou-li zaplaceny všechny položky účtu.

Vytisknutá účtenka má tyto atributy:

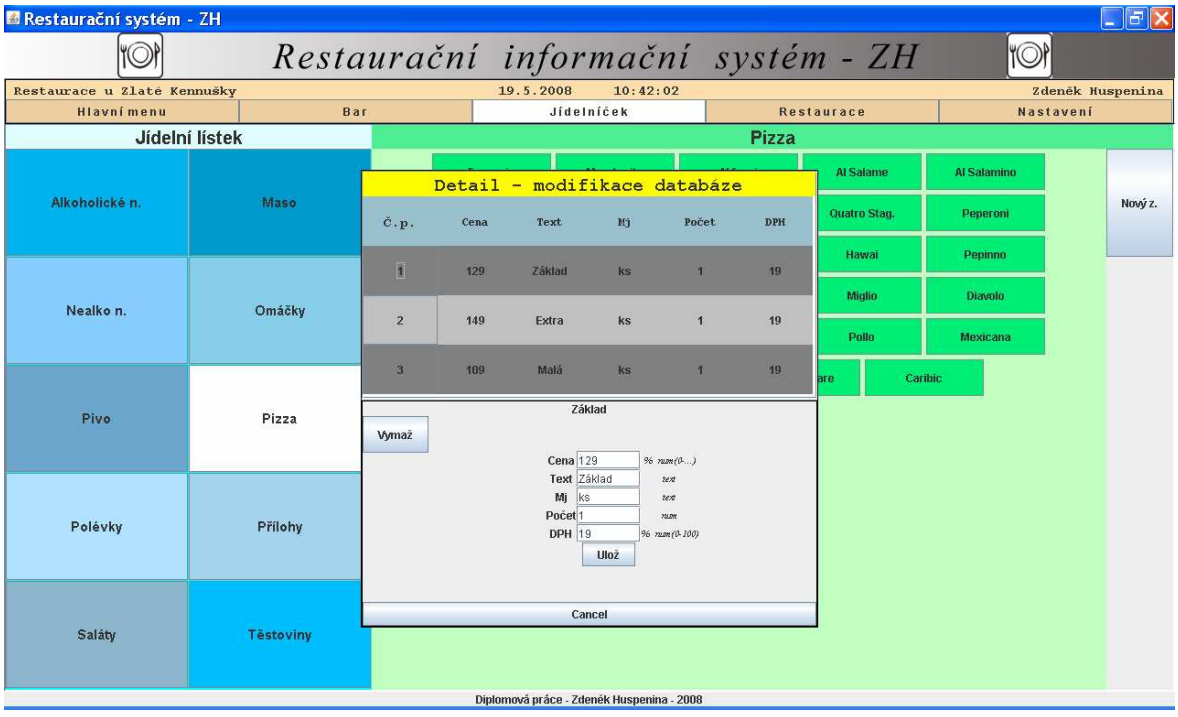
- Záhlaví účtenky – hlavička účtenky obsahuje základní informace o restauraci, například jméno restaurace, adresu a další údaje informativního charakteru.
- Tělo účtenky – tělo účtenky obsahuje informace o jednotlivých položkách příslušné objednávky.
- Zápatí účtenky – zápatí účtenky obsahuje údaje, jež se zobrazují na sumační liště. Mezi tyto údaje patří daň z přidané hodnoty, celkovou cenu a další.

Rozměr vytisknutého účtu byl aplikován přímo na rozměry, jež používá standardní POS tiskárna, avšak tisk proběhl na tiskárně používající standardní formát A4. Při aktivaci tisku byly zablokovány všechny možnosti nastavení tisku, a to z důvodu větší efektivnosti a rychlosti při platebním procesu. Aktivací tisku v informačním systému neprodleně proběhne fyzický tisk účtenky. Nastavení tiskárny lze provést prostřednictvím standardních nástrojů používaného operačního systému.

Příklad vytisknuté objednávky, která neobsahuje stálého zákazníka a tudíž ani žádnou slevu lze najít v příloze PI.

7.5 Modul Jídelníček

Modul jídelníček implementuje složitější grafické uživatelské rozhraní a aplikační logiku pro práci s nabízenými produkty. Prostřednictvím tohoto modulu může obsluha systému pořizovat nové a modifikovat již vytvořené položky jídelníčku. Hlavní logikou modulu je strukturalizace jídelníčku do jednotlivých logických celků. Logický celek přestavuje sekci jídelníčku, například přílohy, nealkoholické nápoje, minutky a další. Každá sekce pak obsahuje produkty, jež mají specifický vztah obvykle k jejich názvu, například hranolky, minerální voda, smažený sýr atd. GUI jídelníčku je rozděleno do dvou vertikálních oken (panelů). Levý panel v sobě implementuje tlačítka s názvy sekcí, které mají funkci zobrazení produktů v pravém panelu. Pro přehlednost má každé tlačítko odlišnou barvu. Právě aktivovaná sekce má bílou barvu pozadí. Názvy hlavních sekcí se pořizují v modulu nastavení, respektive v submodulu sekce.



Obrázek 19. – Modul Jídelníček – modifikace produktu

7.5.1 Vytvoření položky produktu v jídelníčku

Uživatel si musí nejdříve vybrat sekci, kterou bude aktualizovat. Sekce se vybere kliknutím na funkční tlačítko nacházející se v levém panelu modulu jídelníček. Při vytvoření nové položky jídelníčku musí uživatel kliknout na funkční tlačítko „Nový

záznam“, což má za následek otevření popup okna s dalšími možnostmi práce. Okno obsahuje seznam již pořízených detailů a vstupní pole pro zadání všech důležitých atributů.

Mezi tyto atributy patří:

- Název – tato položka musí být vyplněna. Představuje text, jenž bude zobrazen ve funkčním tlačítku daného produktu.
- Popis – krátký popis produktu. Volitelná položka.
- Poznámka – případná poznámka. Volitelná položka.

Vstupem pro všechny tyto atributy je libovolný textový řetězec.

Dalším krokem je potřeba vytvořit detail (funkční tlačítko „Detail“ v popup okně). Detail obsahuje bližší specifikaci k příslušnému produktu. Informační systém rozeznává produkty podle specifikace a to prvního a druhého (viz. Obrázek 15.). Jestliže je vstupní pole číselného formátu a uživatel zadá na vstup textový řetězec, systém zablokuje psaní textu do pole. Jedná se tedy o kontrolu vstupních informací již na úrovni vyplnění. Běžné systémy používají zpravidla kontrolu vstupu až při odeslání formuláře na zpracování. Všechny atributy detailu jsou povinné, tzn., musí být vždy vyplněny ve vstupních polích.

Bližší specifikace (detail) má následující atributy:

- Cena – Určuje cenu s DPH za produkt příslušné specifikace, jenž lze zadat pouze ve formátu čísel. Například: 100.
- Text – Identifikační informace detailu. Údaj s možností libovolného textu. Například: malá porce.
- Mj – Měrná jednotka detailu. Údaj informativního charakteru, vstupem může být libovolný text. Například: g (jako gram).
- Počet – Informace souvisí s atributem Mj. Realizuje numerický údaj počet jednotek. Například: 400.
- DPH – daň z přidané hodnoty v procentech. Například 19 %.

7.5.2 Modifikace produktu v jídelníčku

Každou položku jídelníčku lze libovolně upravovat. Kliknutím na požadovaný produkt v pravém panelu modulu jídelníček se otevře modifikační popup okno.

Modifikační okno obsahuje naprosto totožné atributy, jako okno při vytvoření nové položky produktu v jídelníčku. Uživatel se musí řídit pokyny jednotlivých atributů. Stejným způsobem lze upravovat i bližší specifikaci daného produktu. Aktivace modifikačního okna pro úpravu detailu se provede funkčním tlačítkem „Detail“ a volbou čísla položky v seznamu existujících detailů aktuálního produktu. Jestliže chce uživatel vymazat produkt nebo detail produktu musí aktivovat funkční tlačítko „Vymazat“. Pro ověření procesu odstranění dat, je uživatel dotázán formou varovného okna s možností neprodleně ukončit proces vymazání nebo potvrzení výmazu. Pokud uživatel maže položku na úrovni produktu, provede i výmaz všech jeho detailů.

7.6 Modul Restaurace

Modul restaurace obsahuje aplikační logiku a grafické uživatelské rozhraní pro práci s již pořízenými objednávkami. Jedná se o výpis všech druhů objednávek z databáze. Druhy objednávek byly strukturalizovány na uzavřené, otevřené a všechny objednávky. Z důvodu snadné orientace a přehlednosti bylo grafické uživatelské rozhraní rozděleno do záložkových komponent. Každý druh objednávek představuje jednu záložku. Záložky se též někdy nazývají karty. Horní část záložek s nadpisem dané záložky pak ucha. Slouží k přepínání mezi jednotlivými druhy. Každá záložka má svůj specifický význam a vlastnosti a obsahuje privátní aplikační logiku. Tento modul lze spustit z modulu „Hlavní nabídka“ nebo z lišty s funkčními tlačítky umístěné v záhlaví hlavního okna systému, kliknutím na tlačítko „Restaurace“.

Obrázek 20. – Modul Restaurace

7.6.1 Záložka uzavřené objednávky

Jsou objednávky, které jsou již uzavřeny, z toho důvodu nelze pořizovat a modifikovat její položky a atributy. Uzavření dané objednávky je realizováno při procesu platby účtu. Jestliže objednávka již neobsahuje žádnou nezaplacenou položku, systém automaticky objednávku uzavře a aktualizuje interní atribut čas, kdy byla objednávka uzavřena. Záložka uzavřené objednávky vypisuje pouze objednávky, jež byly uzavřeny v právě aktuální den, protože výpis ostatních objednávek by způsobil nepřehlednost při prohlížení objednávek. Objednávky se při výpisu řadí podle jednotlivých stolů. Kliknutím na funkční tlačítko se zobrazí popup okno s GUI pro prohlížení všech relevantních údajů o objednávce. Kliknutím na volbu „Cancel“ se ukončí prohlížení dané objednávky.

7.6.2 Záložka otevřené objednávky

Obsahuje seznam všech objednávek, které jsou ještě otevřeny, tzn. jejich položky nejsou zcela zaplaceny. Aplikační logika systému je realizována tak, aby uživatel mohl vytvářet libovolný počet objednávek, proto musí existovat záložka pro otevřené objednávky, které lze dále upravovat. Objednávky jsou řazeny podle jednotlivých stolů, z důvodu snadné a rychlé orientace obsluhy systému. Jestliže chce uživatel pokračovat

s prací na příslušné objednávce, musí kliknout na funkční tlačítko dané objednávky. Kliknutím se otevře objednávka v modulu Bar.

7.6.3 Záložka všechny objednávky

Tato záložka je zpřístupněna pouze privilegovaným uživatelům s daným oprávněním. Uživatel tak může prohlížet všechny doposud uzavřené objednávky v časové závislosti. Záložka „Všechny objednávky“ má tedy informativní charakter o všech objednávkách a jejich položkách. Kliknutím na danou objednávku se otevře popup okno s informacemi o dané objednávce.

7.7 Modul Nastavení

Modul nastavení plní funkci administrace systému. Struktura modulu nastavení je rozdělena do čtyř funkčních částí. Každá z těchto částí slouží pro administraci příslušné databázové tabulky. Protože se jedná o modifikaci velmi důležitých dat, je každá z těchto částí privilegovaná. To znamená, že uživatel musí mít dané oprávnění k přístupu do požadované funkční části. Zobrazené grafické uživatelské rozhraní se generuje pomocí naprogramovaného algoritmu, jenž rozliší danou funkční část, a v závislosti na její atributy pak GUI zobrazí. Jeden algoritmus obhospodařuje tedy všechny čtyři funkční části.

Funkční části modulu nastavení:

- Uživatelé systému
- Zákazníci
- Stoly
- Sekce jídelníčku

Obrázek 21. – Modul Nastavení

Pro vstup musí privilegovaný operátor kliknout na funkční tlačítko požadované části. Kliknutím se zobrazí seznam již pořízených dat. Nyní má uživatel možnost vytvořit nový nebo modifikovat již existující záznam. Nový záznam se pořídí kliknutím na funkční tlačítko „Nový“ nacházející se zcela nalevo v GUI okna. Poté se aktivuje popup okno s formulářem pro vyplnění atributů. Chce-li uživatel modifikovat existující záznamy, klikne na funkční tlačítko, které se nachází u každého záznamu ve sloupci označující č.p. číslo položky (číslo záznamu). Zobrazí se opět popup okno s vyplněnými informacemi o daném záznamu, uživatel tak může upravovat tyto informace podle předepsaných pravidel pro každé vstupní textové pole. Další možností je vymazat libovolný záznam v dané funkční části. Odstranění záznamu se provede pomocí funkčního tlačítka „Vymaž“ nacházející se ve formuláři popup okna. Před odstraněním je uživatel ještě dotázán, zdali chce záznam opravdu vymazat, z důvodu nechtěné ztráty dat.

7.7.1 Uživatelé systému

Uživatelé (obsluha systému) jsou osoby, které pracují se systémem. Tato funkční část slouží jako vstupní opatření pro modul autentifikace. Definuje jednotlivé uživatele a jejich oprávnění k přístupu do privilegovaných sekcí. Uživatele systému by měl zpravidla pořizovat management daného restauračního zařízení, protože má největší pravomoce.

Atributy pro uživatele systému:

- Uživatelské jméno – uživatelské jméno, které identifikuje uživatele jako danou entitu. Libovolný textový řetězec. Povinný údaj.
- Heslo – heslo zadané uživatelem. Libovolný textový řetězec. Povinný údaj.
- Ověření hesla – údaj se nezapisuje do databáze, slouží pro kontrolu ověření zadaného hesla před uložením do databáze. Údaje ve vstupním poli heslo a ověření hesla musí být totožné.
- Zákazníci – autentifikační údaj pro vstup do části zákazníci. Booleovská hodnota. Volba realizována formou zaškrtávacího tlačítka. Zaškrknuto – vstup povolen.
- Stoly – autentifikační údaj pro vstup do části stoly. Booleovská hodnota. Volba realizována formou zaškrtávacího tlačítka.
- Všechny obj. – autentifikační údaj pro vstup do části všechny objednávky. Booleovská hodnota. Volba realizována formou zaškrtávacího tlačítka.

7.7.2 Zákazníci

Řešený restaurační IS implementuje slevový systém. Lze tedy pořizovat databázi stálých zákazníků, přičemž každý může mít příslušné cenové zvýhodnění a individuální cenovou politiku.

Atributy pro stálé zákazníky:

- Jméno – identifikační jméno stálého zákazníka. Libovolný textový řetězec. Povinný údaj.
- Sleva – sleva pro stálého zákazníka. Vstup musí být číselného formátu a v intervalu <0,100>, přičemž hodnota znamená slevu v %.

7.7.3 Stoly

Každá objednávka má specifickou vazbu k příslušnému umístění daného stolu v restauraci. Proto je zapotřebí udržovat databázi všech stolů v restauraci. Obsluha systému tak může identifikovat objednávku prostřednictvím umístění.

Atributy pro stoly :

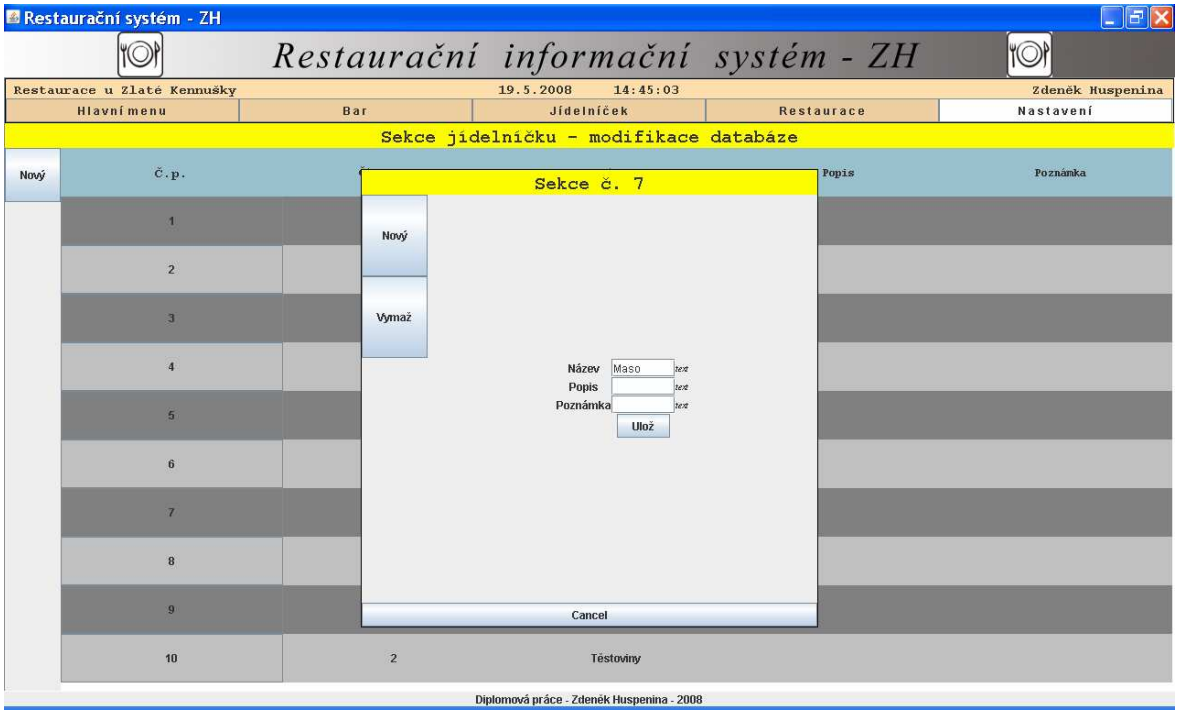
- Umístění – identifikační údaj pro každý stůl. Záleží pouze na pořizovateli záznamů, jakým způsobem bude stoly rozlišovat. Jestliže má restaurace více specifických prostorů, jako například zahrádka, salonek, bar (barové židle), může uživatel zadávat vstupy ve formátu zah_1, zah_2, sal_1 a další.

7.7.4 Sekce jídelníčku

Jídelníček je rozdělen do logických celků, jež se nazývají sekce. Pod pojmem sekce si lze představit například sekci příloh, nealkoholických nápojů atd. Sekce dále obsahuje produkty.

Atributy pro sekce jídelníčku:

- Název – identifikační název sekce. Libovolný textový řetězec. Povinný údaj.
- Popis – udává bližší popis dané sekce. Libovolný textový řetězec. Nepovinný údaj.
- Poznámka – případná poznámka k sekci. Libovolný textový řetězec. Nepovinný údaj.



Obrázek 22. – Modifikace hlavní sekce jídelníčku

8 PŘÍPADNÉ ROZŠÍŘENÍ RESTAURAČNÍHO INFORMAČNÍHO SYSTÉMU

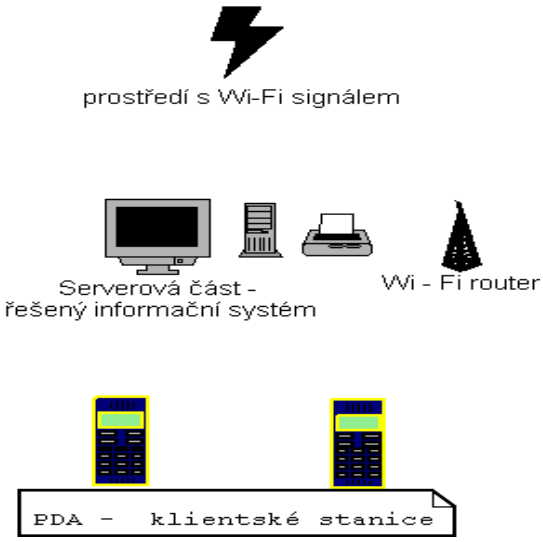
Každý informační systém je zpravidla vyvíjen s příslibem rozšíření o jeho další funkčnost. Téměř pokaždé vydrží dobrá aplikace v běžném provozu mnohem déle, než se původně plánovalo. Za dobu životnosti informačního systému se zpravidla změní okolní podmínky, např. změna legislativy, neustálý vývoj nových technologií a další. Z toho důvodu musí být IS schopen tyto změny akceptovat. S tím se ovšem musí počítat dopředu, již během návrhu a vývoje aplikace. Proto je velmi důležité navrhnout architekturu a strukturu celé aplikace tak, aby byla snadno rozšiřitelná o novou funkčnost s pokud možno co nejmenšími zásahy do stávajících aplikační logiky a tudíž i do zdrojového kódu. Sebemenší změna zdrojového kódu v běžící aplikaci je velmi náchylná k chybám, proto je třeba aplikaci důkladně testovat. Přirozeným vývojem je potřeba zařazovat do aplikace stále novou funkčnost a vlastnosti, proto se může stát, že bude aplikace pomalejší a obtížnější na případnou správu. Při správném návrhu bude probíhat proces snižování výkonu podstatně pomaleji. Záchranou je tedy zvyšující se výkonnost hardwarových prostředků. Životnost žádné aplikace však není nekonečná a může nastat situace, kdy je vhodné znovu zrevidovat celkové požadavky na systém a přepsat celou aplikaci.

Řešený restaurační informační systém lze rozšířit o mnoho nových vlastností a funkcí, záleží na daných požadavcích a na představivosti. Při vývoji systému od samých základů je potřeba stanovit rozsah funkčnosti celého systému, jenž musí bezproblémově fungovat. Potom lze uvažovat o dalších možnostech rozšíření. Podle mého názoru je nejvhodnějším rozšířením systému možnost bezdrátového sběru objednávek a vytvoření grafické mapy stolů.

8.1 Bezdrátový sběr objednávek

Možnost bezdrátového sběru je již ve fázi vývoje, jádro a architektura restauračního IS byla vyvíjena s předpokladem mobilního sběru objednávek. Základní funkčnost již byla otestována, avšak z důvodů velkého rozsahu implementace nebyla tato funkce přidána do koncepce řešení koncového restauračního systému.

Bezdrátový sběr objednávek zcela zásadně rozšiřuje funkčnost restauračního informačního systému. Technické řešení je založeno na architektuře klient-server, přičemž klient je bezdrátové mobilní zařízení, zpravidla PDA, jenž bezdrátově komunikuje pomocí naprogramovaného rozhraní se serverem (řešený systém). Systém je potřeba rozšířit o síťové hardwarové prostředky. V praxi systém funguje tak, že uživatel chodí s PDA po restauraci a zadává objednávky, které se neprodleně po bezdrátové síti zobrazí na serveru (u baru), restaurace tak má možnost zrychlit proces objednávání. Pomocí tohoto způsobu lze minimalizovat chyby v objednávkách, které mohou vzniknout špatnou komunikací mezi obsluhou restaurace a zákazníkem.



Obrázek 23. – Struktura IS pro bezdrátový sběr objednávek

8.2 Mapa restaurace

Další vhodným rozšířením může být mapa stolů dané restaurace. Jedná se o prvek s možností rozložení stolů pomocí grafických prvků. Mapu je možné realizovat pomocí čtvercového rastru, který tvoří půdorys daného restauračního zařízení. Uživatel pak pomocí nástrojů drag and drop a mřížkového rastru vytváří mapu stolů, respektive model restaurace. Při pořízení objednávek obsluha nemusí přemýšlet o textovém významu pro identifikaci stolu, protože stůl k objednávce volí z vizuálního prvku – mapy stolů.

ZÁVĚR

Řešení problematiky objednávkového procesu v běžném restauračním zařízení je rozsáhlou oblastí s velkými možnostmi. Nejefektivnějším způsobem se jeví použití dvouvrstvé architektury tlustého klienta. Architektura celého systému, objektově orientovaná analýza a jednotlivé funkční celky byly navrženy použitím unifikovaného modelovacího jazyka UML. Základním stavebním kamenem celého informačního systému jsou Java technologie a prostřednictvím virtuálního stroje Java se řešená aplikace stává přenosnou a spustitelnou na více operačních systémech, což podstatně zvyšuje její možnosti použití. Celá aplikace je naprogramována programovacím jazykem Java a při vývoji byl kladen důraz na využití všech aspektů objektově orientovaného programování s možností rozšíření aplikace o další funkční prvky. Datové úložiště celého systému je postaveno na nejrozšířenější komerční databázové technologii společnosti Oracle. Z nabízených produktů byl vybrán bezplatný produkt Oracle 10g Express Edition, jenž je poskytován zdarma, avšak s jistým hardwarovým omezením. Technologie Oracle 10g Express Edition byla zvolena z důvodů bezproblémového přechodu na případný placený databázový produkt společnosti Oracle, jenž plně poskytuje technickou podporu a záruku.

Řešený informační systém byl navržen a implementován pro použití v běžném restauračním zařízení. Grafické uživatelské rozhraní bylo realizováno tak, aby ovládání systému bylo jednoduché a uživatelsky přívětivé. Systém byl rozdělen do unikátních funkčních celků (modulů) a každý modul obsahuje specifické GUI a příslušnou aplikační logiku. Společně pak tvoří všechny moduly ucelenou koncepci restauračního informačního systému. Jedním z požadavků na IS byla možnost přihlášení více jednotlivých uživatelů, proto systém implementuje autentifikační modul s rozšířenou funkčností rozpoznání pravomocí každého uživatele. Nejdůležitější vlastností řešeného systému je správná realizace objednávkového procesu v daného restauračním zařízení, proto obsahuje sofistikovanou aplikační logiku objednávek s možností využití příslušných funkcí, jako jsou například systém slev, detailní specifikace nabízených produktů, možnost separátní platby, tisk výsledné účtenky a další. Protože systém obsahuje řadu modifikovatelných databázových položek je rozšířen o možnost jejich administrace. Jako výstup ze systému lze uvažovat vytisknutou celkovou nebo separátní účtenku dané objednávky. Aby mohl informační systém plně konkurovat již existujícím aplikacím je zapotřebí jej rozšířit o další moduly a funkčnost, jako například možnost bezdrátového sběru objednávek nebo

intuitivní volbu umístění příslušné objednávky. Možnost bezdrátového sběru je již ve fázi vývoje, jádro a architektura restauračního IS byla vyvíjena s předpokladem mobilního sběru objednávek. Základní funkčnost již byla otestována, avšak z důvodů velkého rozsahu implementace nebyla tato funkce přidána do koncepce řešení koncového restauračního systému. Po integraci této nadstandardní vlastnosti se stane řešený informační systém efektivnější a objednávkový proces bude rychlejší.

ZÁVĚR V ANGLIČTINĚ

A view into problems of systems covering purchase orders in public boarding facilities shows a very broad field for application of information technologies and opens a great horizons with magnificent technological and marketing opportunities.

For our application, two-tiered architecture based on a thick client appears to stand for the most efficient approach. The entire application's architecture, object oriented analysis and also all the function modules were designed using the UML modelling language. The project was built on foundation of the Java technology, which brings the benefits of a modern platform enabling almost unlimited portability across various operating systems. All presented code is implemented in Java with the emphasis on rich use of design patterns and all positives of object oriented development with the aim to enable easy application extension for required modules in future. We chose the Oracle database, specifically the free version of Oracle 10g Express Edition, as our main data storage. This platform was chosen especially for smooth transition to a commercial version of Oracle when migrating into a commercial environment, bringing also guarantee and full technical support.

The presented information system was developed and implemented for use in a real restaurant. Design of the user interface was focused good serviceability and neat user experience. The entire application is divided into unique function modules, every of them having a specific GUI and appropriate business logic. All together then stand for a rock solid concept of a restaurant information system.

One of the requirements was a possibility of concurrent user access. For that reason, there is a authentication module with an extended capability of user roles. One of the most important parts of the presented system is a correct implementation of the ordering process of specific restaurant. Sophisticated business logic takes care of special kind of orders (e.g. loyalty discounts), detailed specification of the offered meals and other products, a module enabling separate or joint payment for customers, receipt printing capability and others. On the background, there is a very rich administration interface intended for the use of the barkeeper or manager.

In addition to this, the architecture is open and ready for collecting purchase orders using a wireless devices either carried by staff or being available on the customers tables.

This option is currently under development, partially tested but was not included into this work for reason of its complexity and large volume of ongoing implementation. Including this feature, the ordering systems becomes more efficient, the process quicker and hopefully, the customers happier.

SEZNAM POUŽITÉ LITERATURY

[1] *Tenký klient* [online]. [cit. 2008-04-20]. Dostupný z WWW: <http://cs.wikipedia.org/wiki/Webov%C3%A1_aplikace>

[2] KANISOVÁ, H., MÜLLER, M., *UML srozumitelně.*, Computer Press, Brno 2004, ISBN 80-251-0231-9

[3] SPELL, B., *Java programujeme profesionálně.*, Computer Press, Brno 2003, ISBN 80-7226-667-5

[4] POKORNÝ, J., *Úvod do .NET Framework* , Mobil Media, Brno 2002, ISBN 80-86593-16-9

[5] *NetBeans - IDE* [online]. [cit. 2008-03-10]. Dostupný z WWW: <http://www.netbeans.org>

[6] KYTE, T., *Oracle – Návrh a tvorba aplikací, Computer* Press, Brno 2007, ISBN: 80-251-0569-5

[7] *Hibernate* [online]. [cit. 2008-05-12]. Dostupný z WWW: <http://www.hibernate.org>

[8] MCCONNELL, S., *Dokonalý kód – Uměni programování a techniky tvorby software do .NET Framework* , Computer Press, Brno 2005, ISBN: 80-2510849-X

[9] ZAKHOUR, S., HOMMEL, S., ROYAL, J., RABINOVITCH, I., RISSER, T., HOEBER, M., *Java 6 – Výukový kurz.*, Computer Press, Brno 2006, ISBN: 978-80-251-1575-6

[10] KISZKA,B., *1001 tipů a triků pro programování v jazyce Java* , Computer Press, Brno 2005, ISBN: 80-7226-989-5

[11] *Developers – Sun* [online]. [cit. 2008-04-11]. Dostupný z WWW: <http://developers.sun.com>

[12] PALETA,P., *Co programátory ve škole neučí aneb Softwarové inženýrství v praxi*, Computer Press, Brno 2003, ISBN: 80-251-0073-1

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

GUI	Graphic user interface – grafické uživatelské rozhraní
RDBMS	Relational database management system – systém pro řízení relačních dat
(X)HTML	HyperText Markup Language - značkovací jazyk pro hypertext
AJAX	Asynchronous JavaScript and XML – asynchronní JavaScript a XML
XML	Extensible Markup Language - rozšiřitelný značkovací jazyk
CSS	Cascading Style Sheets – kaskádové styly
DOM	Document Object Model - objektový model dokumentu
CASE	Computer-aided software engineering - počítačová podpora SW inženýrství
UML	Unified Modeling Language - grafický jazyk pro návrh a vizualizaci SW
IDE	Integrated development environment - integrované vývojové prostředí
JVM	Java Virtual Machine – virtuální stroj jazyka Java
API	Application programming interface - rozhraní pro programování aplikací
JRE	Java Runtime Environment – prostředí pro chod Javy
JDK	Java Development Kit – vývojové nástroje pro Javu
JSP	JavaServer Pages – Java serverové stránky
EJB	Enterprise JavaBeans - komponenty pro distribuované systémy
JDBC	Java Database Connectivity - rozhraní pro přístup k relačním databázím
RMI	Remote Method Invocation – volání vzdálených metod
RAD	Rapid application development- systém pro rychlý vývoj aplikací
SOAP	Simple Object Access Protocol - protokolem pro výměnu zpráv - XML
SQL	Structured Query Language - standardizovaný dotazovací jazyk
IS	Informační systém
HW	Hardware
DPH	Daň z přidané hodnoty

SEZNAM OBRÁZKŮ

SEZNAM PŘÍLOH

PŘÍLOHA P I: TISK ÚČTENKY

Restaurace u Zlaté Kennušky

ÚČTENKA 523
Datum a čas 19.05.2008 16:44

Restaurace u Zlaté Kennušky , s. r. o.
Chaloupky 574, Veselí n. Mor
69801
IČ: 2323451

Produkt	Specifikace	D[%]	Počet	Cena
Jim Beam	0,02 l	19	3	90,0
Fanta	0,33 l	19	3	66,0
Kuřecí říž.	Velký	19	3	294,0
Hranolky	Malá p.	19	3	60,0
Kozel 12°	0,5 l	19	3	69,0

Celkem: 579,0 CZK

DPH [%]	19
Základ	469
Daň	110

Účtoval: Zdeněk Huspenina
E-mail: info@uzlatekennusky.cz
Web.: www.uzlatekennusky.cz

Děkujeme za Vaši návštěvu

Diplomová práce - Zdeněk Huspenina

PŘÍLOHA P II: DDL SQL TABULKY A TRIGGERU DETAIL

```
CREATE TABLE "SYSTEM"."DP_DETAIL"
  (    "ID" NUMBER, "JID" NUMBER, "CENA" NUMBER NOT NULL ENABLE,
       "TEXT" VARCHAR2(200 BYTE) NOT NULL ENABLE,
       "MJ" VARCHAR2(5 BYTE), "POCET" NUMBER, "DPH" NUMBER,
       PRIMARY KEY ("ID") USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
            STORAGE(
            INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645 PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT
            )
       TABLESPACE "SYSTEM"  ENABLE,
       FOREIGN KEY ("JID") REFERENCES "SYSTEM"."DP_JIDELNICEK" ("ID") ENABLE
  )
PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255 NOCOMPRESS LOGGING
STORAGE(
       INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
       PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT
)
TABLESPACE "SYSTEM" ;
```

```
CREATE OR REPLACE TRIGGER "SYSTEM"."TRIG_DP_DETAIL"
BEFORE INSERT ON DP_DETAIL
FOR EACH ROW
BEGIN
SELECT SEQ_DP_DETAIL.nextval into :new.id from dual;
END;
/
ALTER TRIGGER "SYSTEM"."TRIG_DP_DETAIL" ENABLE;
```