VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

VÝDEJNÍ ČÁST STRAVOVACÍHO INFORMAČNÍHO SYSTÉMU

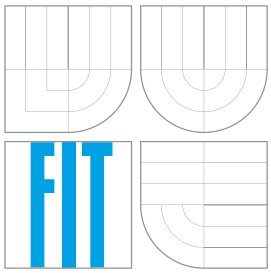
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. VÁCLAV JONSZTA

BRNO 2007

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

VÝDEJNÍ ČÁST STRAVOVACÍHO INFORMAČNÍHO SYSTÉMU

SERVICE POINT PART OF THE CATERNG INFORMATION SYSTEM

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — Bc. VÁCLAV JONSZTA
AUTHOR

VEDOUCÍ PRÁCE — Ing. RADOMÍR KUREČKA
SUPERVISOR

BRNO 2007

Zadání diplomové práce

Řešitel	**Václav Jonszta, Bc.**
Obor	Informační systémy
Téma	**Výdejní část stravovacího informačního systému**
Kategorie	Softwarové inženýrství

Pokyny:

1. Seznamte se s prostředím Microsoft .NET.
2. Seznamte se s vlastnostmi a funkcemi systému ISKaM 2006.
3. Seznamte se s požadavky na funkcionalitu výdejní části informačního systému pro stravování.
4. Zanalyzujte tuto část IS a navrhněte třívrstvou architekturu systému a objektové schéma.
5. Zrealizujte tuto část IS v zadaném softwarovém okolí včetně integrace se stávajícím systémem.
6. Zhodnoťte výslednou implementaci z hlediska reálného nasazení u zákazníka, především z hlediska rizik výpadku některé části systému, bezpečnosti apod.
7. Na základě zhodnocení z bodu 4 navrhněte opatření a rozšíření systému.

Literatura:

- Prosise, J.: Programováni v Microsoft .NET, Computer Press 2003
- Kačmář, D.: Programujeme .NET aplikace, Computer Press 2001

Při obhajobě semestrální části diplomového projektu je požadováno:

- Seznam funkčních požadavků dle bodu 3.
- Analýza systému dle bodu 4.

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

Technická zpráva diplomové práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap, které byly vyřešeny v rámci ročníkového a semestrálního projektu (30 až 40% celkového rozsahu technické zprávy).

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

Vedoucí	**Kurečka Radomír, Ing.**, UIFS FIT VUT
Datum zadání	28. února 2007
Datum odevzdání	22. května 2007

Licenční smlouva

Licenční smlouva v kompletním znění je uložena v archivu Fakulty informačních technologií Vysokého učení technického v Brně.

Výňatek z licenční smlouvy:

Článek 2
Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizování výpisů, opisů a rozmnoženin.

2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.

3. Autor souhlasí se zvěřejněním díla v databázi přístupné v mezinárodní síti:

   ☐ ihned po uzavření této smlouvy

   ☐ 1 rok po uzavření této smlouvy

   ☐ 3 roky po uzavření této smlouvy

   ☐ 5 let po uzavření této smlouvy

   ☐ 10 let po uzavření této smlouvy

   (z důvodu utajení v něm obsažených informací).

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

Abstrakt

Tato práce se zabývá tvorbou stravovacího informační systému. Kvůli velkému rozsahu je zpracována "pouze" výdejní část. Tato WinForm aplikace je postavena na vnitřní třívrstvé architektuře. Z vnějšího pohledu má však jen dvě vrstvy - klientskou a serverovou. Zmíněná výdejní část umožňuje kromě nastavení jídelníčků (a příslušných náležitostí), také práci s dotykovými displeji při výdeji stravy. Velký důraz byl kladen na uživatelské rozhraní, stabilitu a také řešení problémů jako je např. offline provoz.

Klíčová slova

Informační systém, Stravovací informační systém, Menza, Visual Basic, .NET

Abstract

This thesis is about making catering information system. "Only" service point part has been realized, because of huge size of whole system. This WinForm application is based on internal 3-layer architecture. From outer perspective, there are only two layers - client one and server one. Besides customizing bill of fare, service point part also provides efficient work with touch screens, which are used for food output. Strong emphasis lay on implemented user interface, stability and also solving problems like offline functionality.

Keywords

Information system, Catering information system, Visual Basic, .NET

Citace

Václav Jonszta: Výdejní část
stravovacího informačního systému, diplomová práce, Brno, FIT VUT v Brně, 2007

Výdejní část
stravovacího informačního systému

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Radomíra Kurečky. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal. Další informace mi poskytl Ing. Petr Dub.

.....
Václav Jonszta
22. května 2007

Poděkování

Děkuji svému vedoucímu Ing. Radomíru Kurečkovi za poskytnuté rady a vedení. Také děkuji Ing. Petrovi Dubovi, který se aktivně podílí na vývoji dalších částí stravovacího systému, za jeho nekonečnou trpělivost při poskytování odborných rad i pomoci.

Obsah

Kapitola 1

Úvod

V brněnských menzách, tedy alespoň těch, které spadají pod VUT, se denně stravují stovky až tisíce lidí převážně z řad studentů vysokých škol, studentů střední integrované školy a dalších. Dnes si již nikdo z nových studentů jistě nedokáže ani představit, jak to vypadalo v nedaleké minulosti, kdy bylo jídla potřeba dopředu objednávat, kdy bylo za jídlo potřeba zaplatit hotově atp.

Díky bezhotovostním platbám z karty je výdej jídel v menze značně urychlen. Ale jen málokdo si uvědomí, že za tímto zrychlením a usnadněním práce musí stát kvalitní a hlavně stabilní informační systém.

V porovnání např. s ubytovacím systémem je při výpadku systému klíčový naprosto jiný časový interval. Při výpadku ubytovacího systému jsou rozhodující dny. Kdežto u stravovacího systému jde o minuty. Proto takovýto systém musí být velmi robustní a pokud k výpadku dojde, tak by na něj měl být připraven a dokázat jej efektivně zvládnout.

Jsou-li stravovací provozy zatíženy tak obrovským počtem strávníků, jen těžko lze dnešní době zvládnout tento nápor bez informačního systému. Informační systém ale nepřináší jen výhody bezhotovostních plateb. Jsou to třeba i výhody, které nejsou na první pohled vidět, jako je účtování, normování, přehledy, statistiky atp.

Stravovací informační systém si můžeme rozdělit např. na 3 části. První se bude starat o klienty, jejich konta, knihu plateb a všechny další věci týkající se účetnictví. Druhá se postará o správné nastavení cen výdejních podle cen skladových. Ceny se pak budou pružně měnit. O snížení kolísání cen o haléře se postarají fixace cen, které budou udržovat konstantní cenu k danému časovému intervalu. Ke změně tak dojde jen v případě, kdy změna přesáhne definovanou hodnotu. A konečně třetí část bude obhospodařovat co a jak se bude zobrazovat obsluhám pokladen na jejich displejích. A právě tato třetí část je ta, kterou se bude tento dokument zabývat.

Tato první kapitola slouží k zasazení řešené problematiky do širšího kontextu, popisuje současný stav a uvádí cíl práce.

Následující kapitola popisuje teroetická východiska.

Třetí kapitola se zabývá návrhem tříd a tvorbou schématu databáze.

Čtvrtá kapitola popisuje implementaci. Velká část je zde věnována tvorbě uživatelského rozhraní pro obsluhu dotykových displejů.

Závěrečná pátá kapitola hodnotí dosažené výsledky a nastiňuje budoucí směr práce.

V příloze se nachází zadávací dokumentace celého stravovacího modulu a nápověda k programu.

Tato diplomová práce navazuje na Semestrální projekt (SEP), který se zabýval převážně návrhem budoucí aplikace. Návrh je popsán ve třetí kapitole a tato třetí kapitola byla (po úpravách) převzata i do této práce.

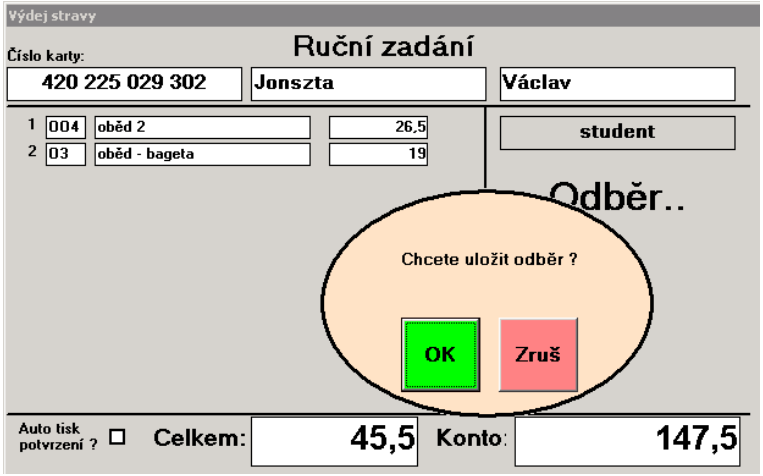
1.1 Současný stav

V současné době je pro provoz menzy využíván informační systém Menza 2000, který byl vytvořen firmou ApS Brno s.r.o. Tento systém byl vytvořen v dnes již zastaralém programovacím jazyce Visual Basic 6.0, který ještě využívá WIN-API funkce. Stávající systém, jak již jeho název napovídá, byl vytvořen v roce 2000 a je zřejmé, že požadavky na stravovací systém byly před sedmi lety o poznání jiné.

V prvé řadě to jsou změny v normování a tvorbě cen. Tyto rozdíly byly největší hnací silou k vytvoření nového informačního systému. Stávající systém totiž umožňuje pouze limitní režim. Limitní režim v praxi vypadá tak, že je dán limit (přesně jako je tomu v brněnských menzách provozovaných KaM VUT, nyní např. 19,00 Kč za normální jídla a 26,50 Kč za jídla výběrová), za který se daná vybraná jídla prodávají. Tato cena ovšem není závislá na surovinách, ze kterých kuchaři jídlo připraví a tudíž není zcela spravedlivá ke strávníkům menzy. Pokud si jeden strávník dá na oběd krupicovou kaši a druhý si dá řízek s bramborami, pak je zřejmé, že nebudou mít oba dva stejné náklady na výrobu svého jídla. Ovšem oba dva zaplatí stejnou sumu.

Za zmínku také stojí to, že stávající systém je po letech již také těžce udržovatelný. Je to způsobeno hlavně postupným nabalováním nově implementovaných požadavků, které nebyly zahrnuty při počáteční analýze. Nové a stále přibývající požadavky na další úpravy tuto skutečnost jen zhoršují.

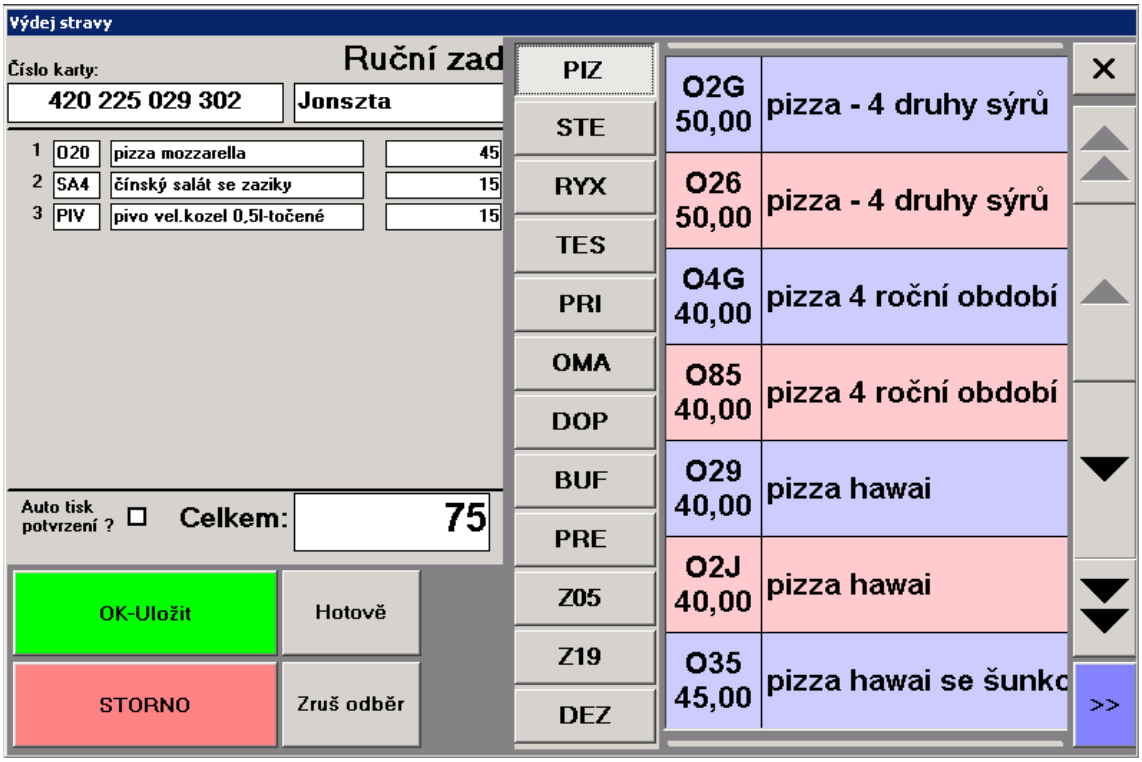
Výdejní část stávajícího informačního systému má také spoustu nevyužité plochy na obrazovkách, které se zobrazují na pokladnách v menzách. Na obrázku 1.1, je vidět obrazovka terminálu bez dotykového displeje, která má jednoduchý a přehledný vzhled a na obrazovkách zabírá jen nepatrnou třetinu celé obrazovky. Nutno podotknout, že nic více ani nepotřebuje, vzhledem k tomu, že počet druhů vydávaných jídel nepřesahuje 10. Ve spojení s klávesnicí s několika málo tlačítky pak účelně napomáhá k rychlému "odbavení" strávníků.



Obrázek 1.1: Stávající obrazovka při výdeji bez dotykového displeje

Výjimku ovšem tvoří menza “Pizzerie Mozzarella”, která využívá dotykových displejů a tím má ovládací plochu rozšířenu o skupinu několika dalších tlačítek. Viz obrázek 1.2. Díky dotykovému displeji již není potřebná klávesnice a veškeré ovládání probíhá přes dotykový displej.

Je zde trošku nešťastně řešeno posouvání na další položky posuvníky v pravé části. Jednoduchá šipka posouvá o jednu položku a dvojitá šipka na konec resp. začátek. Pokud bychom tedy měli v nabídce např. 20 druhů pizz plus každou bychom měli v nabídce ještě i s krabicí, pak budeme mít 40 položek v záložce pizzy. Dostat se pak na nějakou, která je uprostřed, vyžaduje minimálně 13 kliknutí, než se dostaneme na 20. položku. Další nevýhodou je nemožnost opravy zadávání. Pokud se obsluha pokladny splete u zadávané položky, pak musí zrušit celý odběr a zadat všechny položky znovu. To ale nemusí být až tak velkým nedostatkem, vzhledem k tomu, že většina odběrů čítá méně než pět položek.



Obrázek 1.2: Stávající obrazovka při výdeji s dotykovým displejem

1.2 Cíl práce

Celý systém je modulární. To znamená, že spolupracuje s několika vytvořenými moduly, které jsou spolu provázány díky jádru celého systému. Moduly jako takové nemusí být nainstalovány všechny. Stačí jen ty, které zákazník potřebuje.

Jádro

Lépe celou situaci popisuje obrázek 1.3. Zde je úplně dole vidět společné jádro, které se kromě bezpečnosti celé aplikace, systémových nastavení a účtů jednotlivých uživatelů stará

Obrázek 1.3: Stravovací modul jako součást celého systému

také o správu klientů a ekonomické záležitosti.

Nad společným jádrem můžeme nalézt již zmíněné moduly, které mohou pracovat samostatně, bez ostatních modulů, nebo i využít jejich provázanosti.

Hotel

Lze tak snadno využít existující konta klientů z hotelového modulu v modulu stravovacím. Většina strávníků, kteří se v menzách stravují jsou studenti a většina z nich zároveň bydlí na kolejích, takže můžeme snadno ve většině případů využít společného konta.

V praxi to bude mít ty následky, že když ubytovaný nezaplatí ubytování, tak se taky nenají v menze. Tím odpadá problém informování neplatičů, kteří nečtou nástěnky s informacemi o nezaplacení za ubytování. Zároveň to také umožňuje zaplatit ubytování v menze nebo si při zaplacení za ubytování připlatit více a z toho pak čerpat při stravování v menze.

Sklady

Sklady evidují nejen suroviny na skadu, ale také hotové výrobky na "výdejní lince" (speciální sklad), odkud jsou následně odepisovány podle toho, co se namarkuje ve stravovací části.

Poměrně důležitým procesem, který tedy ve skladech probíhá, je výroba - tj. vytvoření "výdejky do výroby" a k ní odpovídající "příjemky z výroby", kde prvním dokladem se vyskladní z nějakého skladu suroviny a druhým se obecně na jiný sklad přijmou hotové výrobky. Musí ale platit, že celková cena je na obou dokladech stejná, takže cena surovin se přímo promítá do ceny hotových výrobků, což je základ pro bezlimitní ceny ve stravování.

Sklady mohou pracovat v jednom ze dvou režimů.

- FIFO - Cena jednotlivých surovin je dána tím, za jakou cenu byly tyto suroviny naskladněny. Např. máme na skladě naskladněných 100kg brambor. První padesátka byla naskladněna za cenu 20,- Kč (za kilogram) a druhá za 25,- Kč. Nejprve budou

vyskladněny brambory za cenu 20,- Kč za kilo a po vyskladnění 50kg se cena změní na 25 Kč za kilogram.

- Průměrná cena - v tomto režimu budeme mít podle předchozího příkladu na skladě 100kg brambor za cenu 22.50,- Kč.

Normování

Normování je nadstavba nad *sklady*, která usnadňuje a do značné míry automatizuje vytváření výše zmíněných dokladů. K tomu potřebuje tzv. receptury, neboli kolik a jakých surovin je potřeba pro výrobu 100 porcí určitého jídla. "Norma" je potom konkrétní implementací této receptury, tj. zadá se, kolik porcí se má skutečně vyrábět (systém samozřejmě přepočítá "trojčlenkou", kolik surovin bude potřeba), ze kterého skladu vzít suroviny, kam naskladnit hotové výrobky a datum výroby.

V jeden den se pro určitou dvojici zdrojového a cílového skladu dělá více produktů, takže vygenerovaná výdejka a příjemka je pak pro všechny tyto výrobky společná. Normování však musí správně promítnout cenu surovin do těch produktů, kde jsou použity (v závislosti na jejich množství).

Dále produkuje normování podklady pro kuchaře, kde mají napsáno, ze kterých surovin mají udělat to které jídlo.

Stravování

Poslední částí je modul *stravování*, který slouží pro usnadnění práce ve stravovacích provozech. Využit bude převážně v menzách, ale ani použití v jiných provozech není vyloučeno.

Oproti stávajícímu systému, který pracuje v limitním režimu, bude nový ssystém umožňovat práci navíc i v bezlimitním režimu. Ceny mohou být přímo závislé na skladových cenách, nikoliv na limitech (pevně definovaných hodnot). K tomuto ovšem budou potřebné moduly *sklady* a *normování*.

Původní sytém byl postaven pouze pro bezobjednávkový provoz. Narozdíl od toho bude nový systém umět pracovat v jednom z těchto 3 režimů:

- Objednávkový - strávníci si pomocí připravených terminálů vybírají jídla na několik dní dopředu. Pokud později zjistí, že je již nechtějí, mohou je dát do "burzy". Někdo jiný si toto jídlo může vzít i v den výdeje, čímž získá jídlo, které by si jinak musel objednávat několik dní dopředu a původnímu majiteli takto nepropadnou peníze za jídlo, které si sice objednal, ale nesnědl.

- Bezobjednávkový - žádné objednávání jídel dopředu není potřeba. Strávníci si jídla vybírají přímo v době výdeje.

- Kombinovaný - bude kombinovat předchozí dvě možnosti.

Cílem této práce je vytvořit výdejní část tohoto nově vyvíjeného modulu. Stejně jako celý systém, bude i tento modul postaven na (v současné době moderní) technologii .NET. Kvůli rozšíření o bezlimitní způsob tvorby cen, již nebude existovat jen malé množství předdefinovaných cen (např. 1 za normální jídlo a 1 za výběr), ale každá položka, která se bude vydávat bude mít svou vlastní cenu. Tímto přibývá možnost lépe využít dotykové displeje ve výdejnách.

Jak již bylo uvedeno dříve, k chodu stravování jsou důležité 3 části. První se stará o konta klientů a ekonomické záležitosti. Toto obhospodařuje jádro celého systému. Druhá

část se stará o cenotvorbu, vzorce atp. a poslední třetí část obhospodařuje výdej a vše co je s ním spojené.

U prvních dvou částí záleží dokonalý chod na správném prvotním nastavení. V průběhu dalšího provozu pak již bude docházet jen k občasným úpravám a tedy k nepříliš velkému vytížení těchto částí. Jinak je tomu u výdejní části. S výdejní částí budou uživatelé na příslušných místech pracovat nepřetržitě několik hodin denně na několika provozech současně. Proto musí být při vývoji této části kladen navíc důraz na několik dalších důležitých aspektů.

Vedle stability a bezpečnosti, které jsou důležité pro všechny části, je zde potřeba klást navíc hlavně důraz na rychlost aplikace. Ta je zde klíčová a bez dostatečné rychlosti by byla aplikace nepoužitelná i kdyby byla sebevíc stabilní a pěkná. Druhým velmi důležitým aspektem je uživatelská přívětivost. Právě vzhledem k tomu, že budou uživatelé nuceni pracovat se stále stejně vypadající obrazovkou, je potřeba zajistit aby byla aplikace co nejvíce podle představ uživatelů. Není však možné se zavděčit všem, proto maximalní variabilita, umožňující nastavení nejrůznějších vlastností, ať chování či barev, bude na místě. Každý si tak bude moci nastavit vše podle vlastní potřeby.

Stejně tak je potřeba vyřešit neduhy předchozího systému. Hlavně bude potřeba vyřešit tzv. *offline* provoz, který umožní i nadále vydávat jídla a zadávat odběry do systému i v případě kdy dojde k přerušení spojení se serverem.

Kapitola 2

Teoretická a odborná východiska řešených problémů

2.1 Obecný systém

Systém je účelově definovaný soubor komponent, mezi kterými existují určité vazby, a které splňují nějaký cíl. Skládá se z atributů, událostí a časových množin. Atributy jsou veličiny charakterizující určitý prvek systému; události představují změnu atributu nebo změnu konfigurace systému a časové množiny jsou hodnoty vztažené k určitému okamžiku.

Systémová analýza se zabývá systémy vytvořenými lidmi, jež se skládají ze vstupů, procesů a výstupů.

Hranice systému určuje samotný systém nebo odděluje více systémů. Pomyslnou hranicí je logická hranice, která vymezuje podsystémy v rámci systému, zatímco "viditelnou hranicí" je již **okolí systému**. Chování systému pak ovlivňují prvky nacházející se zvenku hranice.

2.1.1 Rozdělení systémů

Systémy obecně dělíme na **tvrdé** a **měkké**. Tvrdé systémy jsou spojeny s jedním specifickým problémem, zatímco u měkkých systémů vystupuje celá řada faktorů, jsou obecnější. Systémy lze dále členit na:

- *uzavřené* nebo *otevřené* - podle toho, zda systém komunikuje s okolím či nikoliv. Je však nutné podotknout, že uzavřené systémy v reálném světě neexistují, protože téměř každý systém je jistým způsobem spojen s okolím;
- *deterministické* nebo *stochastické* - tj. jednoznačné nebo statistické chování,
- *statické* nebo *dynamické* - tj. lineární nebo diferenciální,
- *spojité* nebo *diskrétní* - podle časových událostí.

2.1.2 Zpětná vazba

V systémech může nastat zpětná vazba, kdy výstupní veličina opětovně ovlivňuje vstupní veličinu, a tudíž i samotný systém. Zpětná vazba představuje důležitou vlastnost systému. Každý systém má tendenci být nestabilní, a proto může pomoci zavedení tzv. regulátorů.

2.2 Informační systémy

Informační systémy, jak je již z názvu patrné, jsou systémy pracující s informacemi a daty. Informační systémy jsou určeny pro sběr, udržování, zpracovávání a poskytování informací a dat, jsou také nezastupitelným nástrojem, jež výrazným způsobem usnadňují a urychlují práci, zvyšují efektivnost práce a zlepšují kvalitativní charakteristiky podniku či organizace. Informačním systémem může být kartotéka, telefonní seznam nebo účetnictví. Systém nemusí být automatizovaný pomocí počítačů, může mít i papírovou podobu. S informačními systémy souvisejí tyto základní pojmy: informace, data a znalosti.

- **Informacemi** se rozumí sdělení odstraňující nejistotu či nevědomost. Informace je údaj, ke kterému si člověk přiřadí určitý význam.
- **Daty** se rozumí jakékoliv získané poznatky nebo fakta.
- **Znalosti** představují zobecnění poznání určité části reality.

Obecně se tedy informační systémy definují jako systémy pro zpracování dat. Data mohou mít tyto cíle:

- strategické,
- taktické,
- operační.

Podstatné jsou rovněž úlohy informačních systémů, které mohou být:

- manažerské (EIS – Executive IS),
- taktické (DSS – Decision Suppor System),
- vedení (MIS – Management IS),
- expertní (KWS - Knowledge Work System),
- kancelářské (OIS - Office IS),
- operativní:
  - TPS - transakční (např. banky),
  - CRM - péče o zákazníka,
  - RIS - rezervační systémy,
  - CAM - konstrukční (např. CAD),
  - GIS - geografické systémy.

2.2.1 Rozdělení IS

Informační systémy se dělí podle čtyř kritérií:

- podle informačního prostředí – některá prostředí jsou si podobna, tudíž jsou podobné i IS. Vznikají pak tedy tzv. typové prostředí jako jsou účetnictví nebo knihovny,

- podle organizační úrovně řízení – využívá se hierarchického členění organizací. Např. od velkých koncernů až po jednočlenné firmy,
- podle převládající funkce IS,
- podle režimu činností.

2.3 Projektování IS

2.3.1 Projektování

Organizaci řízení tvorby a návrhu systému můžeme rozdělit na několik fází:

- úvodní studie,
- rozbor zadání,
- analytické modelování,
- systémový design,
- objektový design,
- implementace,
- zkušební provoz,
- a nasazení.

2.3.2 Navrh

Specifikace a návrh aplikace jsou silně ovlivňovány tím, jaká bude architektura systému. Návrh obsahuje popis modelovacích technik, které jsou použity pro lepší abstrakci skutečného systému. Hlavním artefaktem jsou tedy případy užití nebo jinak také modely jednání (use cases). Mezi základní prvky patří: aktér, scénář a impuls-reakce (zpráva). Případy užití je možné, obdobně jako v softwarovém inženýrství, rozšiřovat nebo generalizovat.

Na základě případů užití vzniká **model spolupráce**. Úlohou tohoto modelu je hledání prvních náznaků tříd, odpovědností a vztahů. Výsledky těchto činností pak ústí v objektový model, který již přesně zachycuje celý systém, vztahy mezi objekty či hierarchii dědění.

Kontrolní pohled na vytvářený systém poskytuje **funkční model**. Standardem je zde DFD (Data Flow Diagram), který umožňuje snadné grafické vyjádření propojitelné s datovým modelem. DFD diagramy obsahují:

- aktéry (obdélník - například osoba, instituce, jiný systém a podobně),
- datové sklady (obdélník se zaoblenými rohy bez pravé strany - uchovává data),
- procesy (obdélníky se zaoblenými rohy - manipulují s daty, jsou algoritmy) a
- datové toky (šipky - předávání datových záznamů).

DFD model je hierarchický, což znamená, že procesy se dají postupně zjemňovat. Každý proces obsahuje "vnořený" diagram, a tak dále až po takzvané listové procesy, jež jsou atomické (nedělitelné). Každý proces v DFD obsahuje textový popis (např. pseudokód, přirozený jazyk, různé podmínky, apod.), popis omezení (constraints) a dodatečné informace (jako např. možnosti optimalizace).

K pochopení změn v systému přispívá **dynamický model**. Možné popisy jsou například: slovní scénáře, grafické scénáře (např. sekvenční diagramy), mapy událostí (jeden diagram na celý systém) nebo stavové diagramy a tabulky. Samostatnou kapitolou jsou pak ER-diagramy, které zachycují datový model.

2.3.3 Architektura

Významným aspektem je volba architektury. Téměř výhradně se používá třívrstvá architektura:

- prezentační vrstva (interakce s uživatelem),
- funkční vrstva (vlastní aplikace, bezpečnost, propojení se světem, kontrola, atd.),
- datová vrstva (komunice s DB, soubory atp.)

Významná je i bezproblémová integrace IS, která má dvě hlediska - **vnitřní** a **vnější**. O vnitřní hledisko se jedná, když dochází k proškolení pracovníků, nastavení prostředí a podobně, zatímco u vnějšího se jedná především o zákazníky a dodavatele.

Převzato z [Wi107].

2.4 .NET

.NET je zastřešující název pro soubor technologií v softwarových produktech, které tvoří (nebo mají tvořit) celou novou platformu Microsoftu od serverů přes webové XML služby, které jsou základem .NET, až po klientské systémy (tzv. chytré klienty).

Chytrý klient umí využívat XML služeb .NET a má tedy umožněn stálý přístup k informacím. Webové služby jsou jednotlivé (typicky krátké) programy, které data buď zprostředkovávají, nebo umožňují jejich sdílení.

Pro tvorbu aplikací splňujících tyto myšlenky vydal Microsoft Visual Studio .NET, které bylo oproti předchozí verzi rozšířeno o snadný návrh webových XML služeb .NET, a .NET Framework, zajišťující prostředí potřebné pro běh aplikací a nabízející jak spouštěcí rozhraní, tak potřebné knihovny.

.NET není vázán pouze na jeden programovací jazyk, ale umožňuje psát kód v několika různých programovacích jazycích např.:

- Visual Basic
- C#
- J#
- a další

Informace použité v této kapitole byly částečně převzaty z wikipedie ([4], [5])

Kapitola 3

Návrh

Dle zadání je potřeba nejprve vytvořit diagram případů použití, který poslouží pro názornější pochopení zadání. Zadání je podrobně rozepsáno v zadávací dokumentaci v příloze A.

Dále bude následovat návrh tříd a následně schéma databáze.

3.1 Případy použití

Jak zobrazuje digram případů použití 3.1 s výdejní částí budou pracovat dva typy aktérů:

- Pokladní - osoba, která bude pracovat v menze u dotykového displeje a zadávat do systému složení jednotlivých odběrů jídel. Bude moci nechat nabít uživateli vloženou částku na konto, případně mu na to vydat příslušný doklad.
- Provozní - osoba, která bude mít pravomoce nastavovat, jaká jídla budou k dispozici, jak budou organizována a jaká bude nabídka jídel na daný den.

Editací se zde míní celá kolekce CRUD. Vytváření nových položek, čtení, úpravy stávajících objektů a mazání (Create, Read, Update, Delete). Mazání je ovšem omezeno pouze na nevyužité položky. Nelze tedy např. smazat komoditu (jídlo), která je na další den naplánována v jídelníčku. Naopak, pokud vytvoříme novou komoditu, která zatím není nikam naplánována do jídelníčku, pak její smazání není problém.

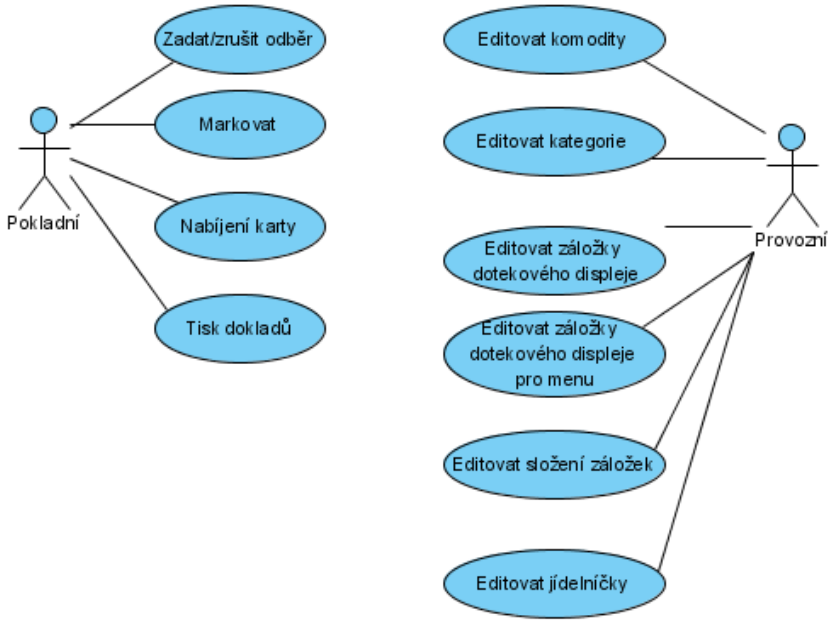
3.2 Návrh tříd

3.2.1 Komodity

Základem veškerého výdeje i celé menzy jsou komodity, proto tedy začneme právě komoditami. Komoditou je označováno cokoliv, co je vyprodukováno k prodeji. V našem případě to bude vše, co má přidělenu prodejní cenu. Není problém, aby na jednom provozu byla komodita *řízek s bramborami* a na druhém provozu byly komoditami *brambory* a *řízek* každá zvlášť. Vše záleží na vnitřním nastavení aplikace.

Původní program využíval pouze jednoduché komodity. Tento základní koncept rozšíříme o komodity složené nebo-li tzv. *menu*. Jednoduchá komodita bude nejmenší základní položka, která se v menze vydává (např. guláš). Složená potom bude např. “Sýrová pizza v krabici”, která se bude skládat z pizzy a z krabice.

Na *menu* mohou stravovací zařízení vydávat speciální slevy. *Menu* se tedy bude skládat např. ze dvou částí. Z polévek a hlavních jídel. Na výběr, tak budeme mít několik polévek



Obrázek 3.1: Diagram případů použití

a několik hlavních jídel, podle denní nabídky. Z každé kategorie bude možnost vybrat si maximálně jedno jídlo. Nic nám samozřejmě nebrání v tom, dát si jedno hlavní jídlo s polévkou jako "Menu1" a druhé hlavní jídlo, které už nebude součástí *menu*(bez zvýhodněné ceny).

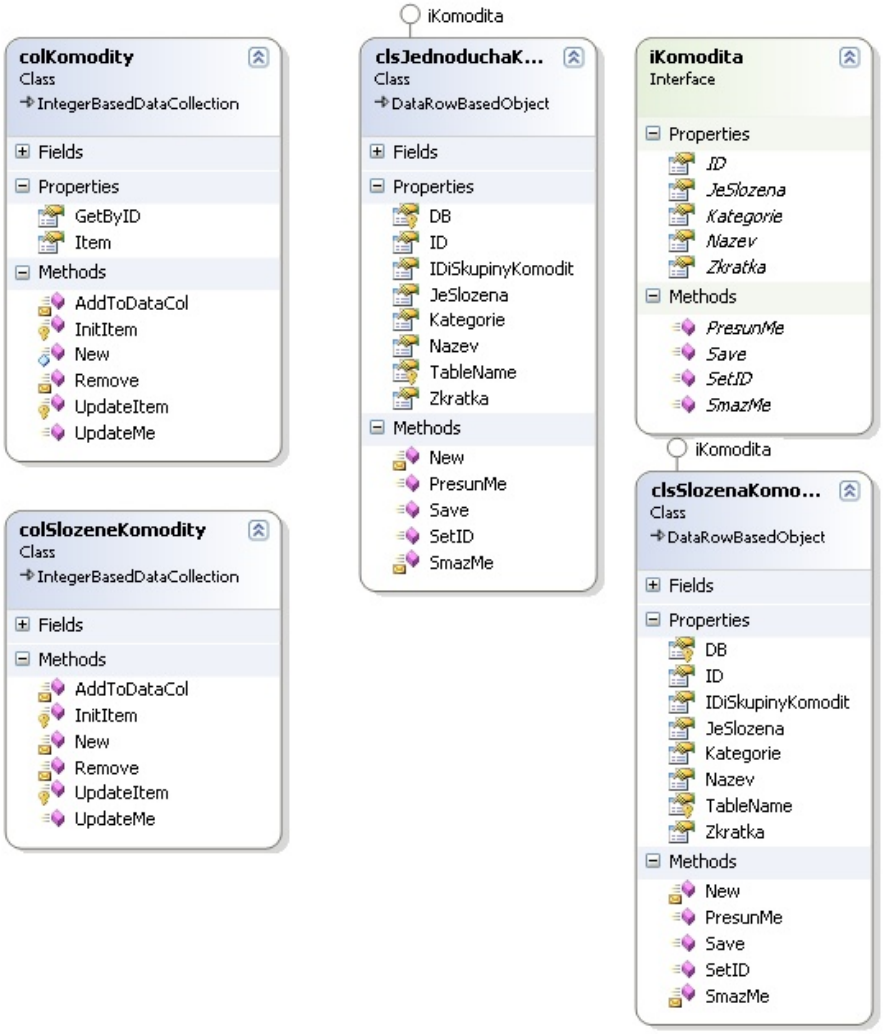
Lépe to vystihuje obrázek tříd komodit 3.2. Je na něm vidět, že komodity vychází z rozhraní *IKomodita*, které definuje některé základní prvky, které každá komodita musí bezpodmínečně mít. Složená komodita se podle vlastností od jednoduché moc neliší, ovšem bude mít rozdílnou implementaci těchto metod. Také je vidět, že složená komodita neobsahuje žádný atribut, který by zastřešoval kolekci jednoduchých komodit. Je to kvůli tomu, že složení Menu budou časově závislá. Tedy že v pondělí může mít *menu1* jiné složení než *menu1* v úterý.

Na provozech, kde se denní nabídka jídel každý den mění, se přirozeně budou měnit i složení menu (pokud budou použita). S těmito menu můžeme pracovat dvěma způsoby:

- Na každý den definovat nové menu - každé menu bude mít unikátní název a složení.
- Definovat např. jedno menu, které bude mít sice každý den jiné složení, ale název zůstane stejný pro všechny dny.

První přístup je sice snazší pro pochopení, ale kvůli narustajícím seznamům menu bude velmi špatně udržovatelný. Proto použijeme druhý způsob.

Využijeme také kolekce jednoduchých i složených komodit pro jednodušší práci s komoditami.

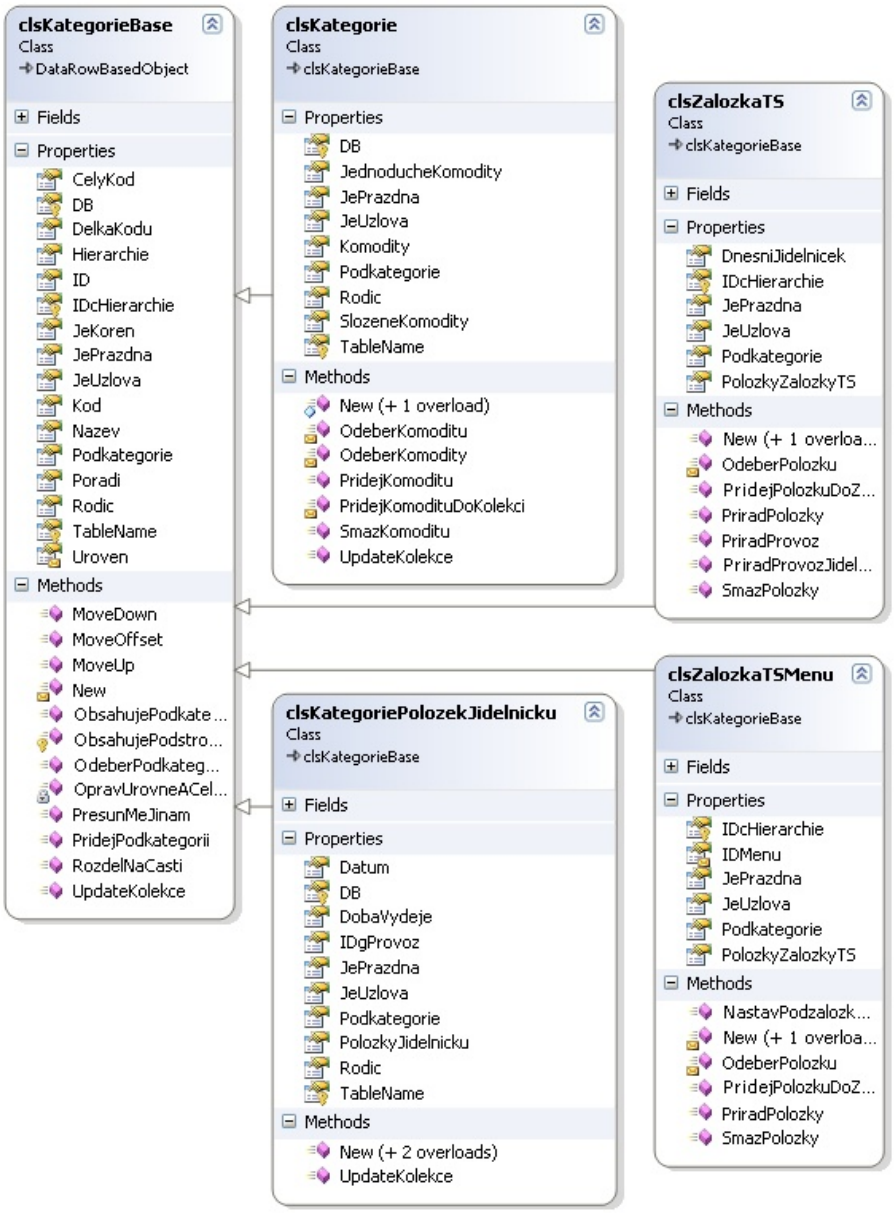


Obrázek 3.2: Diagram tříd - komodity

3.2.2 Kategorie

Abychom se v té spoustě komodit neztratili, bude potřeba je kategorizovat a hierarchicky uspořádat. K tomuto účelu využijeme objektu *Kategorie*, který vychází z objektu *KategorieBase*. *KategorieBase* obsahuje všechny potřebné vlastnosti pro práci se stromovou hierarchickou strukturou, kterou díky této bázové třídě budou moci využít i další hierarchicky strukturované objekty.

Každá kategorie má ovšem jedno omezení. Nelze v ní mít zároveň další podkategorie současně s komoditami. Vždy jen jedna z variant je možná. Komodity lze mít v uzlové kategorii (to je každá, která nemá další podkategorii). Kategorie zobrazuje obrázek 3.3, který kromě kategorie a bázové třídy uvádí další hierarchicky strukturované objekty, jejichž rozbor bude následovat v dalších kapitolách.



Obrázek 3.3: Diagram tříd - kategorie

3.2.3 Záložky dotykového displeje(touch screen)

Záložky poslouží, podobně jako v předchozí kapitole *kategorie*, k lepší organizaci všech vydávaných položek na dotykovém displeji.

Záložek dotykového dipleje využijeme ve dvou verzích. Klasickou *Záložku dotykového displeje* (ZalozkaTS) a *Záložku dotykového displeje pro menu* (ZalozkaTSMenu). Tyto třídy budou, stejně jako kategorie v předchozí kapitole, využívat stromovou hierarchickou strukturu. Záložky se nebudou lišit podle provozu, tzn. že všechny provozy budou pracovat se stejnou kolekcí záložek. Ovšem zobrazovat se při výdeji budou jen ty záložky, které byly na daném provoze naplněny nějakými položkami jídelníčku. Pokud na jednom provozu budeme chtít využít jen první tři záložky, tak pak komodity, které budeme na tomto provozu vydávat, přiřadíme jen těmto třem záložkám.

PolozkyZalozkyTS bude kolekce staticky přiřazených komodit k daným záložkám, tak, jak se mají zobrazovat na dotykovém dipleji. Dynamicky přiřazené komodity na daný den a dobu výdeje(viz. dále) budou v kolekci *DnesniJidelnicek*

U druhé varianty je to poněkud komplikovanější. Záložka menu je totiž o úroveň hlouběji než normální záložka. Pokud např. na pokladně klikneme na jednoduchou záložku v dialogu výdeje stravy, zobrazí se jídelníček složený z jednoduchých i složených komodit. U jednoduchých není co řešit. Při výběru složené komodity (menu) se zobrazí zcela nová nabídka jídelníčku, která je organizována právě do záložek dotykového displeje pro menu.

3.2.4 Položky v záložkách

Jak již bylo uvedeno dříve, na dotykovém displeji budou komodity uspořádány do jednotlivých záložek. Protože však již komodity mají své umístění v hierarchicky struktorovaných kategoriích, použijeme další objekt a to *položku záložky*. Budeme sice využívat většinu vlastností, které mají komodity, ovšem implementace některých vlastností bude jiná. Rozhodující je hlavně skutečnost, že samotné komodity jsou organizované do hierarchické struktury kategorií, což nám neumožňuje použít stejný objekt.

Použijeme tedy *položku záložky dotykového displeje* (clsPolozkaZalozkyTS) pro obyčejné komodity a *položku záložky dotykového displeje uvnitř menu* (clsPolozkaZalozkyTSMenu). Položky uvnitř menu mají opět jinou implementaci, zvláště v tom, že používají navíc ještě jednu tabulku.

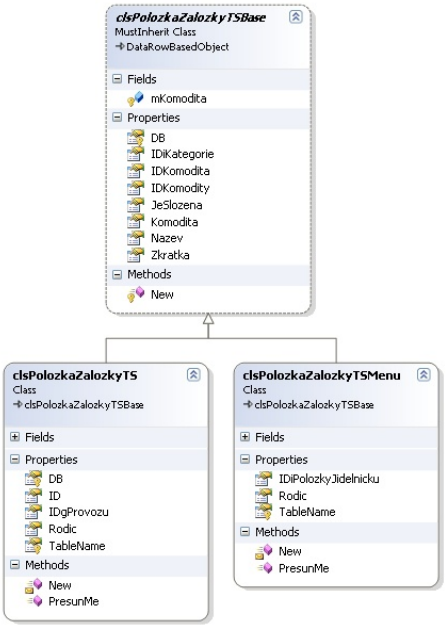
Některé vlastnosti ovšem budou stejné pro obě třídy. Využijeme tedy bázovou třídu clsPolozkaZalozkyTSBase. Celé schéma lépe zobrazuje obrázek 3.4

3.2.5 Doby výdeje

Abychom rozdělili jednotlivé části dne a definovali tak, co se bude vydávat k snídani, co k obědu a co na večeři, budeme potřebovat definovat dobu výdeje. Nemusíme zůstat jen u snídaní, obědů a večeří. Nadefinovat si můžeme dle potřeby i další doby (svačiny atp.). Třídu dob výdeje zobrazuje obrázek 3.5.

3.2.6 Položky jídelníčku

Nyní již máme všechny potřebné předpoklady k vytvoření položek jídelníčku. Každá položka jídelníčku potřebuje mít definovaný *provoz*, na kterém se bude vydávat, potřebuje *den* a *dobu výdeje* ve kterou se bude vydávat a taky potřebujeme vědět, jakou *komoditu* budeme vydávat.



Obrázek 3.4: Diagram tříd - položka záložky dotykového displeje

3.2.7 Odběr

Nyní ve spolupráci s cenami, které jsou různé pro jednotlivé podobjednávky a dále jsou závislé na definovaných vzorcích, skladových cenách a dalších informací (cenotvorba není součástí výdejní části, proto se jí nebudeme dále zabývat), se můžeme pustit do samotných odběrů jídla.

Odběrem se míní vybrané položky jedním strávníkem, které mají být zaúčtovány, když prochází přes pokladnu.

Do knihy plateb se budou ukládat pouze položky jídelníčku první úrovně. To znamená, že dá-li si někdo *menu*, pak se do knihy plateb uloží jen to, že si dal menu a příslušná částka, nikoliv výčet toho, co si v rámci daného menu vybral. To, co si vybral, ovšem nezahodíme, ale uložíme do odběrů spolu s jednoduchými komoditami. Tato informace pak např. poslouží při odpisech ze skladů, statistikách atp.

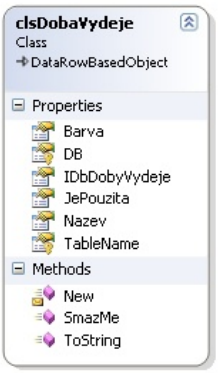
Každý odběr se bude skládat z *položek odběru*, kde každá položka ponese informace o tom, o jakou komoditu se jedná, jestli se jedná o menu (jaké položky tedy byly vybrány), jaká je cena, kolik z toho platí klient (může mít část nebo celou částku hrazenu zaměstnavatelem), zpětnou vazbu na odběr, ke kterému je vázána, atp.

Samotný odběr pak kromě kolekce těchto položek bude obsahovat informaci o tom, o jakého klienta se jedná, tedy kdo je plátcem resp. vlastníkem karty. Dále částku na kontě před i po odběru a jestli již byl odběr ukončen nebo nikoliv.

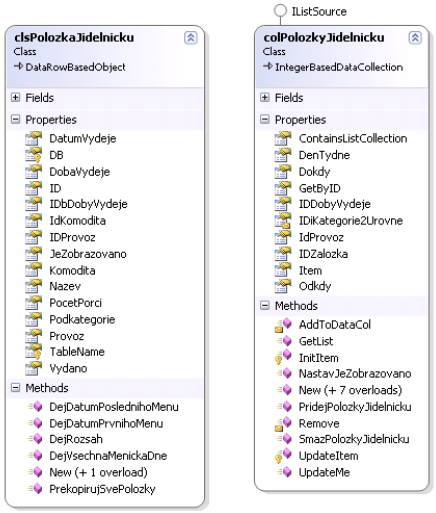
3.2.8 Omezení odběru a příspěvku

Na závěr ještě stojí za zmínku omezení odběru, které je vyobrazeno na obrázku 3.8 spolu s omezením samotným.

Omezení samotné je definováno intervalem (budou definovány pouze tři intervaly - den, týden a měsíc) a pro daný interval můžeme definovat počet odběrů. Pro měsíční interval



Obrázek 3.5: Diagram tříd - doba výdeje



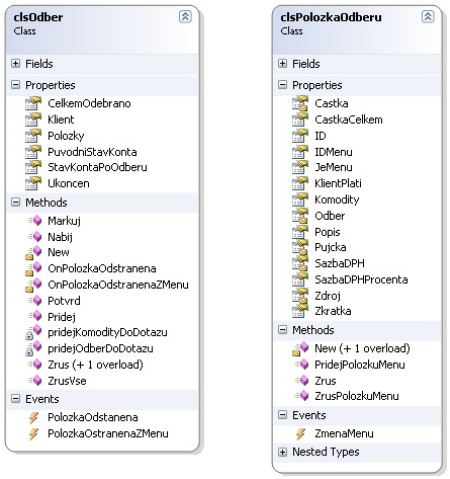
Obrázek 3.6: Diagram tříd - položka jídelníčku

zde navíc bude možnost definovat násobek pracovních dnů, což pak lze snadno využít pro omezení strávníka např. v tom, že může za měsíc sníst jen tolik jídel, kolik je v daném měsíci pracovních dnů. Svátky a víkendy samozřejmě nebudou započítány.

Pro každý interval bude možno vybrat pouze jedno omezení, která jsme si nadefinovali výše. Využít to můžeme např. tak, že omezíme výdej jídel na 5/týden a 2/den. Pokud si tímto omezením omezený strávník dá první dva dny 2 jídla a třetí den sní to poslední na které má ten týden nárok, tak celý zbytek týdne už se nenají. Případně nají, ale za hotové nebo za základní cenu. Toto zatím ještě nebylo specifikováno.

Obdobně se budou chovat také omezení příspěvků. Tato omezení se budou ovšem vázat na samotné vzorce příspěvků a využití najdou převážně u stravování zaměstnanců, kterým na stravu přispívá zaměstnavatel.

Při návrhu architektury, tříd a při tvorbě UML diagramů, bylo využito technik popsaných ve studijní opeře k předmětu Analýza a návrh informačních systémů ([1]).



Obrázek 3.7: Diagram tříd - odběr

3.3 Návrh databáze

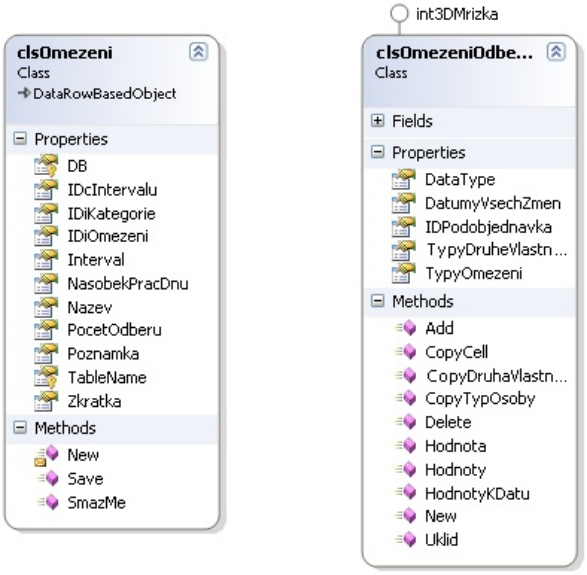
Nejprve si uvedeme trochu terminologie v názvech tabulek. První dvě písmena v názvu každé tabulky označují, že se jedná o

- VY - Výdej
- KA - Katalogy

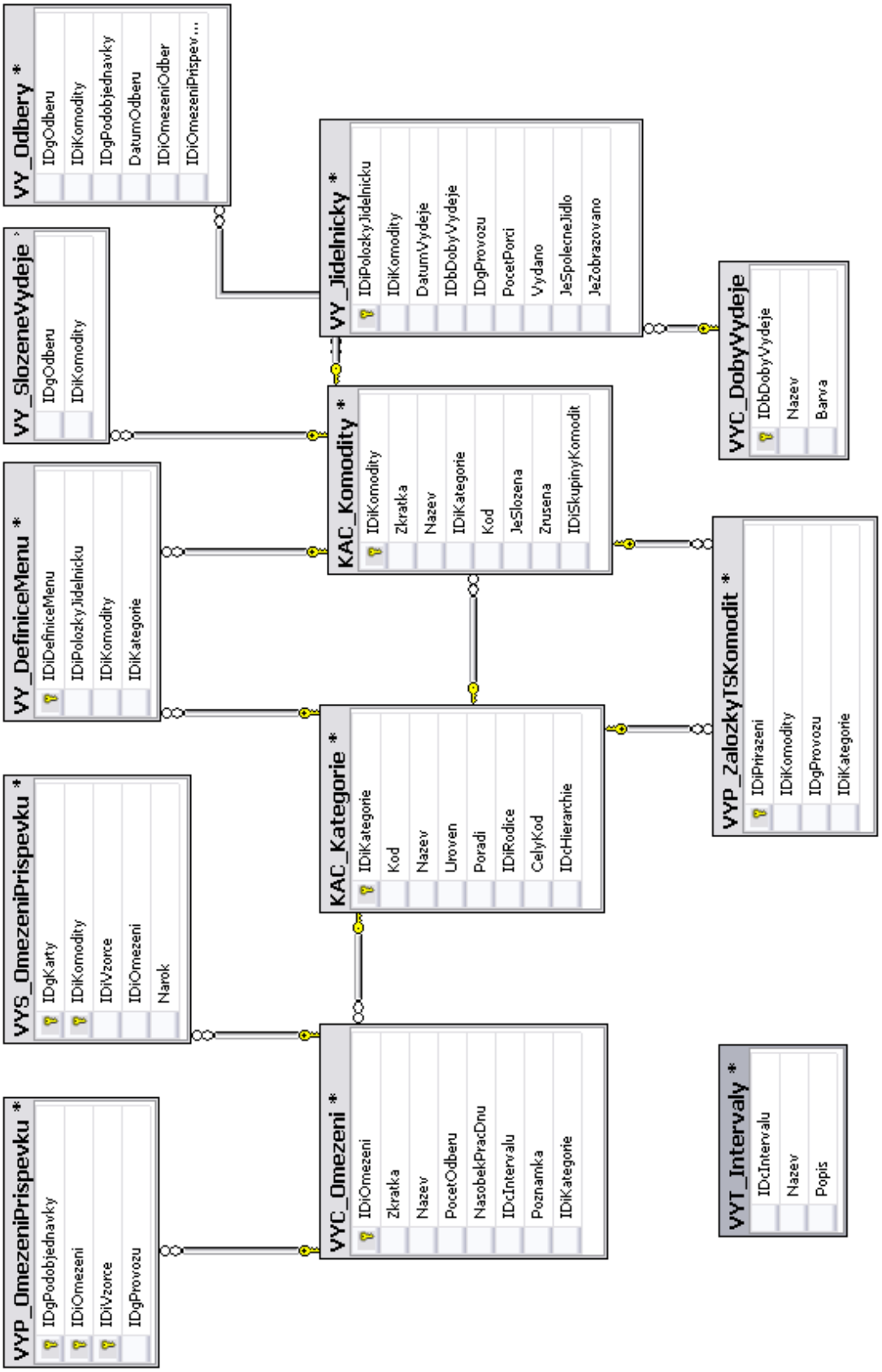
Pokud před podtržítkem leží ještě jedno písmeno, pak se jedná o upřesnění a jednotlivé zkratky pak znamenají:

- C - Označuje uživatelsky rozšiřitelný katalog.
- P - Tato tabulka slouží jako přiřazení položek jedné tabulky k položkám z tabulky druhé.
- S - Takováto tabulka je statická a slouží pro "nakešování" hodnot pro rychlejší přístup k datům, které by se jinak získávaly složitějším a pomalejším dotazem, což by v některých případech vedlo k výraznému snížení odezvy systému.
- T - tato tabulka je definována programově a uživatel její obsah nemůže ovlivnit.

Na schématu databáze 3.9 jsou tabulky pojmenovány v souladu s návrhem tříd a pochopení jejich vzájemného propojení je tedy intuitivní. Více vysvětlují zkratky v názvech tabulek, které byly popsány výše.



Obrázek 3.8: Diagram tříd - Omezení odběru



Obrázek 3.9: Schéma databáze

Kapitola 4

Implementace

4.1 Katalogy

U každé větší aplikace jsou potřebné katalogy, což jsou vlastně seznamy hodnot nebo dat, se kterými budou budoucí uživatelé pracovat a jejichž složení si mohou uživatelé sami definovat.

Při tvorbě programu je vždy potřeba zvážit, jestli bude daný seznam uživatelsky editovatelný či nikoliv. Možnost uživatelsky definovat složení seznamů sice přináší výhody pro uživatele, ale na druhou stranu poskytuje další možnosti výskytu chyb. To platí i všeobecně. Čím více je daná věc flexibilní, tím větší je její šance na chybu či poruchu.

V tomto případě budeme chtít, aby aplikace byla co možná nejvíc flexibilní a přizpůsobitelná a to nejen pro jednu menzu, ale i pro další menzy v jiných městech a případně i v jiných státech. Uživatelskou editovatelnost si můžeme odpustit např. jen v případě časových intervalů jako je den, týden a měsíc. Tyto intervaly jsou využity pro omezení. Ovšem všechny ostatní seznamy budou muset být editovatelné samotným uživatelem.

- Komodity - Toto jsou základní stavební kameny stravovacího informačního systému. Určují co bude možno ve stravovacích provozech vydávat. Obecně takovýchto položek budou stovky. V závislosti na velikosti organizace zajišťující stravování.
- Kategorie - Ty zde poslouží hlavně pro lepší organizaci komodit, pokud jich bude velké množství. Což pro menzy spadající pod VUT rozhodně platí.
- Záložky - Tento katalog poslouží pro organizaci vydávaných položek na dotykových displejích. Přiřazení potom budou různá pro jednotlivé stravovací provozy.
- Doby výdeje - Budou definovat období dne, ve kterých budou výdejny vydávat jídlo. Na tento katalog se poté budou navazovat jídelníčky, které samozřejmě mohou být pro jednotlivé doby různé.

První tři katalogy (Komodity, Kategorie a Záložky) jsou vytvořeny poměrně standardním způsobem a to ve stromové struktuře. Na počátku existuje jen jedna kořenová složka, do které lze intuitivně vkládat další záložky pomocí pravého tlačítka myši.

Posledním katalogem jsou doby výdeje. Tato nenápadná položka je kromě názvu dané doby výdeje obohacena také o barvu, která bude k vybrané době příslušet. Při plnění jídelníčku pak bude na první pohled jasné, který jídelníček, resp. na kterou dobu, se právě plní. Více bude vysvětleno v následující kapitole.

4.2 Jídelníčky

Co jsou to jídelníčky jistě není třeba vysvětlovat. Co ale znamená jídelníček z pohledu tohoto programu, to si vysvětlení určitě zaslouží. Vysvětlíme si to na konkrétním příkladu KaM VUT Brno, kde funguje zároveň několik menz. Některé z nich jsou v provozu jen daný interval, jiné mají provozních intervalů více a tyto provozní intervaly de facto odpovídají dobám výdeje. Pro každou tuto dobu výdeje a příslušný provoz je pak definován seznam jídel, které se budou vydávat.

Mezi jídelníčkem a vydávanými položkami je velký rozdíl. V praxi může být v jídelníčku např. 20 položek, třebaže vydávaných jídel může být v daný okamžik pouze 8 a pokud jsou vyčerpány zásoby jednoho druhu jídla, jednoduše se toto nahradí jídlem jiným.

Vydávané položky jsou v podstatě podmnožinou celého jídelníčku. Jejich nastavení je sice také důležité, ale ne tak, jako nastavení jídelníčku. Samotná nastavení vydávaných položek (jídel), ponesou pouze informaci o podobě tzv. "momentálního jídelníčku". Tento "momentální jídelníček" bude zobrazován strávníkům na LCD televizorech a bude je informovat o tom, která jídla si mohou v daný okamžik dát. Nikde jinde se tato nastavení neprojeví. Teoreticky by samozřejmě mohla, např. na displejích pokladen. Jak si ovšem vysvětlíme později, nebyla by to dobrá volba.

Nyní, když už máme jasno, co je přesně myšleno jídelníčkem, můžeme se podívat na jeho implementaci. Samotná volba toho, jak by bylo možné efektivně plnit jídelníčky, nebyla snadná. Nejjednodušší by samozřejmě bylo nechat obsluhu pro každý den vytvořit seznam jídel, případně zobrazit celý seznam komodit a nabídnout zaškrtávátka. Bohužel žádná z těchto možností není zcela uživatelsky přívětivá.

Implementace jídelníčku vychází z předpokladu, že jídelníčky se plní přibližně týden dopředu. K dispozici máme tedy tabulku, na jejíž x-ové ose máme dny týdne a mezi jednotlivými týdny můžeme samořejmě listovat. A na y-ové ose je seznam jídel, které se alespoň jednou ve vybraném týdnu vyskytují. Obarvení polí v průsečících dne a komodity udává, jestli je daná komodita v jídelníčku nebo nikoliv. Obarvená – ano, bílá – nikoliv. Záložky s dobami výdeje ještě umožňují vybrat správnou dobu a aby se předešlo omylu změny jiné doby výdeje, je zde pomocná právě ta barva přiřazená k době výdeje. Jídelníček pro obědy bude tabulka např. zeleno-bílých políček a večeře půjdou jednoduše rozeznat např. tabulkou modro-bílou. Jednotlivá políčka tabulky jsou pak citlivá na kliknutí, čímž lze přepínat mezi vybráním a zrušením položky z jídelníčku. Samozřejmostí jsou také výběry více položek najednou, prodloužení výdeje položky do konkrétního data a stejně tak i zkrácení.

Zvláštní položkou jídelníčku je menu. Jak již bylo popsáno v návrhu, tak menu je vlastně také položka, která má definovanou svou cenu. Při jejím odběru bude sice zapsána do knihy plateb úplně stejně jako libovolná jednoduchá komodita, ale ze skladu se bude muset odepsat více položek, než u té jednoduché. Navíc jejich složení může být pokaždé jiné. Menu, resp. složené komodity, se v tabulce zobrazují mírně tmavší barvou, která je vypočtena automaticky z barvy doby výdeje. Jejich složení lze naplnit pouze zde v jídelníčku. Pokud je menu ještě prázdné, je označeno křížkem.

4.3 Výdej stravy

Hlavní částí celého stravování je podpora výdeje stravy. Zjednodušeně řečeno se jedná o formulář, který bude zobrazen na dotykových displejích pokladen a s jehož pomocí budou moci pokladní snadno markovat jednotlivé odběry.

4.3.1 Technické požadavky

Jak již bylo uvedeno v předchozích kapitolách, systém bude pracovat v bezlimitním režimu. Nyní, dokud pracuje ještě v limitním režimu, jsou specifikovány za jídlo pouze 2 ceny pro studenty a zaměstnance a další 2 pro cizí strávníky. Pro snadné a rychlé markování položek tedy stačí klávesnice s přibližně 10 klávesami. Větší klávesnice, která by obsluhovala desítky položek, by však ztratila na efektivnosti a možnost jejího dalšího využití bohužel odpadá.

Jako druhá možnost se standardně nabízí myš. Ta má oproti ovládání klávesnicí několik podstatných výhod. Kromě toho, že nezabere tolik místa jako klávesnice s počtem kláves, odpovídajícím počtu jídel, tak umožňuje jednodušeji a hlavně rychleji zadávat jednotlivé odběry do počítače.

Bohužel ani zrychlení získané použitím myši není dostačující. Během špičky bývají v menzách sáhodlouhé fronty. Někdy i více než 100 lidí a v některých menzách bývá v provozu dokonce i jen jedna pokladna. V takových případech je důležité, aby markování bylo co nejrychlejší.

Pokud chceme rychle a efektivně zadávat do systému odběry jednotlivých strávníků, budeme jednoznačně potřebovat dotykové displeje. Cenový rozdíl mezi myší a dotykovými displeji je sice velice výrazný, ovšem výsledná efektivita bude větším přínosem.

Druhým technickým požadavkem je čtečka karet. Jen steží by mohl být výdej plynulý, pokud by nebyly použity čtečky elektronických karet. Načítat konta ručně by celý proces velice zpomalilo. I když na druhou stranu, nelze se jen spoléhat na čtečku karet. Ke kontu tak musí být snadný přístup i při ručním zadání čísla karty. Tento případ se sice bude objevovat jen ojediněle, ale i tak musí mít otevřenou cestu.

4.3.2 Programové požadavky

Programovými požadavky se rozumí to, jaké vlastnosti by měl splňovat výsledný program neboli daná část informačního systému. Tyto požadavky samozřejmě platí i všeobecně, ale zde si je pro úplnost zopakujeme. Jsou to:

1. Ergonomie
2. Strukturovanost
3. Přehlednost
4. Rychlost
5. Efektivita
6. Uživatelská přívětivost
7. Jedoduchost

V následujícím textu se nyní pokusíme rozebrat jednotlivé vlastnosti. Podrobně se podíváme na to, co každá z vlastností znamená a jak jí bylo dosaženo.

4.3.2.1 Ergonomie

Zní to možná zvláštně, ale i program, tedy alespoň ten dobrý, musí vykazovat dobré ergonomické vlastnosti. Při tvorbě uživatelského rozhraní je tedy nutné brát ohled na ergonomii. Je potřeba zajistit, aby uživatele, kteří s programem pracují, nebolely po práci ruce, nebolely je oči atp. Obzvláště to platí u výdeje stravy, kde uživatelé celou pracovní směnu musí sedět u počítače a zírat do obrazovky.

Při návrhu vzhledu bylo využito přínosných zkušeností stávajících uživatelů a také sledování jejich práce. Byly zde sice návrhy na jiné rozložení než u původní aplikace, ovšem ukázalo se, že základ rozložení je pro práci ideální. Navíc jsou uživatelé na toto rozložení již zvyklí a tudíž extrémní experimenty nepřicházejí v úvahu.

Nejčastěji jsou mačkána tlačítka s názvy položek. Při předpokladu, že většina pracovníků obsluhy kasy jsou praváci, pak nejergonomičtější umístění těchto položek musí být v pravé části obrazovky. Při jejich umístění do levé části by si uživatelé rukou zakrývali ostatní položky a celý odběr by se tímto zpomalil. V tento okamžik bohužel aplikace není přizpůsobena pro leváky a neumožňuje změnu stran.

Informativní část je pak přidružena k levému okraji. V této části pak můžeme najít informace o strávníkovi a jeho kontu. Také se zde nachází seznam již vybraných jídel v rámci aktuálního odběru. Na rozdíl od pravé části, kterou se položky přidávají, zde lze kliknutím položky opět odebrat. Nepočítá se však, že frekvence rušení položek bude taková, jako u přidávání. Tudíž nevadí, že jsou tyto položky umístěny v levé části, kde při případném výběru si uživatel zakrývá rukou zbytek obrazovky.

Poslední neméně důležitou částí jsou tlačítka pro potvrzení, nabití, ruční markování a zrušení odběru. Ty jsou umístěny v levé dolní části. Nezabírají tak místo v pravé části, které je potřebné pro seznam položek, kterých se obecně předpokládá velké množství. Umístění v levé dolní části je nejen z důvodu zvyku na toto místo, ale také z toho důvodu, že nejpravděpodobnější pohyb bude: v pravé horní části bude vybráno jídlo, při zapnuté rotaci (bude vysvětleno v sekci u efektivity) se po výběru narotují další položky a po zadání celého odběru pak bude ruka směřovat vlevo dolů k tlačítku potvrdit. Při tomto pohybu není namáháno vůbec rameno. Dochází jen k rotaci v lokti. Navíc to nabízí možnost podepření lokte ruky.

4.3.2.2 Strukturovanost

V případě malého výběru sortimentu, který je dostupný při výdeji, není strukturovanost nutností. Obsluha se v malém počtu položek snadno orientuje a strukturovanost je poté spíše kontraproduktivní a navíc zabírá důležité místo na pracovní ploše.

Záložky

Zcela jiná situace je pak ve výdejnách, kde počet druhů jídel, která se vydávají, se pohybuje v desítkách kusů, nebo až kolem sta. V tomto případě se bez důkladné strukturalizace neobejdeme. O zajištění strukturovanosti se nám postará standardní vztah záložka – položka. Záložka je nadřazena položce a může jich obecně obsahovat více najednou. Jako záložku si můžeme představit například “polévky” a jako položky pak “čočková” a “gulášová”. Sloupec se záložkami je umístěn ve střední části, vpravo od něj jsou sloupce s položkami jídelníčku a vlevo se nachází místo pro markované položky. Záložky by z ergonomického hlediska mohly být spíše úplně vlevo. Rozdíl ve výkonnosti by však nebyl tak významný,

aby si kvůli této změně museli uživatelé zvykat na jiný způsob ovládání, než na jaký jsou již zvyklí.

Základní chování bez záložek by tedy umožňovalo zobrazení všech položek abecedně seřazených bez jakéhokoliv uspořádání podle druhu jídla. Při rozšíření základního modelu o záložky získáme navíc panel s již zmíněnými záložkami. Při vybrání libovolné záložky se zobrazí jen ty položky, které do ní logicky náleží. I když logika vlastně spíše záleží na osobě zodpovědné za příslušná nastavení.

Stránky

Dalšího rozšíření budeme potřebovat, pokud nastane alespoň jeden z následujících dvou případů. Prvním z nich je situace, kdy máme v jídelníčku alespoň jednu záložku, která má více položek než je schopno se vejít na jednu stránku. Nebo v druhém případě máme zapnutý buben, což znamená, že se položky plní za sebou na stránku, i když nepatří do stejné záložky.

V obou případech budeme potřebovat způsob, jak se dostat na položky, které se již nevešly na zobrazenou stránku. Pokusíme se zároveň vyhnout implementaci, která provázela předchozí aplikaci. V té se na další položky dalo dostat pomocí šipek. Tyto šipky ovšem výběr posouvaly pouze o jednu položku.Takovéto chování samozřejmě není přijatelné a proto využijeme mnohem přívětivějšího řešení a tím bude stránkování. V dolní části pod položkami je prostor, kde se zobrazují čísla stránek. Jejich výběrem se výběr zobrazených položek posouvá hned o počet položek, rovný násobku nastavení počtu řádků a počtu sloupců. Toto číslo však není zcela přesné, bylo by potřeba připočítat ještě režii za vypisování nadpisů. Co to jsou nadpisy si vysvětlíme později.

4.3.2.3 Přehlednost

Další důležitou vlastností je přehlednost. Bude-li uživatelské prostředí nepřehledné, pak bude uživateli trvat dlouhou dobu než se v tom "zmatku" zorientuje a to opět snižuje efektivitu práce. To samozřejmě nechceme a pomocí několika vylepšení pomůžeme aplikaci k lepší přehlednosti.

Barvy

První nápomocnou vlastností je střídání barev položek. Tyto jsou pak mnohem lépe odlišitelné jedna od druhé. Oči pak mohou mnohem rychleji a pohodlněji najít hledanou položku. Samotná implementace je také velmi snadná, pokud pracujeme jen s jedním sloupcem položek. Aplikace však dokáže zobrazovat položky až v pěti sloupcích. Máme-li nastaveno zobrazení více než jednoho sloupce, pak se barvy musí střídat nejen v řádcích, ale i sloupcích. Pro každou z hodnot 1-5, představující počet sloupců, bychom pak potřebovali jiný algoritmus pro správné vykreslení střídání barev. Lépe nám však poslouží následující vzorec:

```
((i - 1) Mod pocetSloupcu + Math.Ceiling(i / pocetSloupcu) + 1) Mod 2
```

kde i - je sekvenční číslo určující číslo buňky. V případě nastavení 3 sloupců bude mít políčko na druhém řádku a v druhém sloupci i rovno hodnotě 5. *Mod* - je funkce modulo, která vrací zbytek po dělení a *Math.Ceiling(x)* je funkce vracející hodnotu x zaokrouhlenu směrem nahoru.

Nadpisy

Další pomůckou jsou nadpisy. Ne vždy musí být v záložkách velké počty položek. Zajisté budou vždy existovat provozy, kde v každé záložce bude obrovský výběr, ale stejně tak vedle nich budou i provozy, které nebudou mít výběr tak veliký. Vzhledem k tomu, že aplikace umožňuje spojit obsahy záložek, což znamená, že v pravé části s položkami nebudou jen položky, spadající pod vybranou záložku, je nezbytně nutné tyto položky od sebe nějakým způsobem odlišit. Tím se vyhneme problému smíchání všech položek dohromady.

Nadpisy tedy slouží k oddělení položek jedné záložky od druhé a jsou vždy zobrazeny na samostatném řádku.

Rozsahy

Nadpisy tak dokáží velice snadno a efektivně zpřehlednit celou pravou část. Uživateli však můžeme vylepšením základní verze nadpisů pomoci ještě více. Jedná se o zobrazení rozsahů. Rozsahy jsou užitečné pouze v případě, že se všechny položky výdeje nevejdou na jednu obrazovku. V případě jejich aktivace se zobrazují na stejném řádku jako nadpis a umístěny jsou bezprostředně za ním. Informují uživatele o tom, je-li na stránce zobrazen veškerý obsah záložky, či nikoliv. Pokud opravdu dojde k tomu, že položky přesahují na další stránku nebo dokonce stránky, pak se za nadpisem zobrazí číselný rozsah popisující, které položky na stránce jsou zobrazeny a z jakého celkového počtu.

Např.: Pokud by měl provoz v nabídce 25 pizz a zobrazování by bylo nastaveno na zobrazení jen 10 položek na stránku, pak při vybrání záložky Pizzy by se na první stránce zobrazilo 9 prvních pizz plus nadpis s rozsahem (1-9/25). Obsluha, která potřebuje namarkovat Zeleninovou pizzu, pak na první pohled ví, že ji najde až na třetí stránce a ušetří tak čas hledáním zeleninové pizzy na stránce druhé.

Aktuální záložka

Předposlední vlastností, která také napomáhá v lepší orientaci a přehlednosti a která je známa spíše z webových aplikací, je označení aktuální záložky. Není to složitá věc, jde v podstatě jen o změnu zobrazení tlačítka záložky tak, že vypadá zamáčknutě a jeho text je zobrazen tučně. Horší je to pak s implementací takovéhoto zamáčknutí, pokud se změní aktuální záložka jinak, než kliknutím na ni. Toto nastává v případě automatické rotace. Podrobněji bude tato vlastnost vysvětlena až v sekci, zabývající se efektivitou.

Aktuální stránka

Poslední vlastností je označení aktuální stránky. Jedná se o podobný princip jako u předchozího bodu, jen v kontextu se stránkami.

4.3.2.4 Rychlost

Tato vlastnost patří mezi ty nejdůležitější. Není-li to právě rychlost, která je úplně nejdůležitější, pak mezi ty nejdůležitější alepoň patří. I kdyby aplikace nebyla dokonale přehledná a kvalitně strukturovaná, nebyl by to takový problém, jako kdyby byla pomalá.

Při vývoji došlo k několika problémům s výslednou rychlostí a právě proto byl původní model předělán do nové a rychlejší podoby.

Načítání jídelníčku

Prvním faktorem, který zpomaloval práci u předchozí aplikace, byl fakt, že se celý seznam s jídelníčkem načítal vždy až po započetí odběru. U každého jednoho strávníka vzniklo takto zpoždění, způsobené komunikací se serverem. Extrémní důsledky to mělo u nasazení do menzy, ve které navíc nebylo zcela ideální internetové připojení. Z času zaúčtování jednoho odběru, který čítá přibližně 5s, se stal dvojnásobek a toto zpoždění již nebylo únosné. Výsledkem byla finanční ztráta vlastníka programu.

Nabízí se naprosto triviální řešení a to udržovat jídelníček v paměti a nemuset jej načítat stále znovu. Toto řešení však má jeden háček. Nesplňuje požadavek (možná nesmyslný), zaručující možnost kdykoliv během dne změnit cokoliv v jídelníčku, názvech jídel, atp. Při načítání po každém odběru je toho sice dosaženo, ale za vysokou cenu v podobě zpoždění. Řešením této situace se nabízí kombinace těchto dvou vlastností. Jídelníček je načten pouze jednou při spuštění aplikace, zároveň na pozadí probíhá kontrola, zda-li nebyl jídelníček nebo jeho část změněna.

Pro uchování jídelníčku poslouží soukromá kolekce, která je odlehčená od všech nepotřebných informací a v paměti tak bude zabírat jen malou část. Tato kolekce slouží jako tzv. zdroj dat neboli *datasource*. Obsahuje jak jednotlivé položky, tak i nadpisy. Jednotlivé položky nesou hlavně informace o názvu. V případě položek(tedy ne nadpisů) i jejich ID, které poslouží pro jednoznačnou identifikaci markované položky. A nakonec jsou zde informace usnadňující výpočet rozsahů.

Tyto položky jsou v kolekci seřazeny podle seřazení záložek. Vždy nejprve abecedně seřazený obsah jedné záložky a pak pokračují záložky další. Pro vykreslování správného obsahu už stačí jen určit správný offset do této kolekce. Při generování záložkových tlačítek je informace o offsetu dané záložky v *datasource* uložena do *tag*u příslušného tlačítka. Tím je informace velice dobře a rychle dostupná.

Překreslování položek

První a naivní návrh nepočítal s žádnými problémy při vykreslování. Problémy ale nastaly a to převážně v pomalém vykreslování skupiny tlačítek. V reálné situaci to způsobovalo, že po kliknutí na záložku bylo při překreslování nových položek znatelně vidět samotné překreslování. Pro obsluhu by toto chování bylo zcela určitě neakceptovatelné. Je to podobně nepříjemné jako blikající zářivka. Možná i více, protože na zářivku nemusíte hledět celou pracovní směnu.

Z neznámého důvodu však v .NETu nelze správně nastavit double buffering pro vykreslování skupiny tlačítek. Pokud by toto šlo korektně nastavit, snad by pak nebylo vidět ono postupné vykreslování jednotlivých položek, které tak nepříjemně tahá oči. Jednou z možností, jak tuto situaci vyřešit, je vytvořit si matici polí, odpovídající nastavenému počtu sloupců a počtu řádků. Na tuto matici pak budou navazovat dvě kolekce. První z nich bude obsahovat kolekci tlačítek, odpovídající počtu polí v matici. Druhá kolekci labelů, sloužící jako nadpisy. Jejich počet pak bude odpovídat počtu řádků - 1. Nadpis na posledním řádku nemá smysl vykreslovat, protože už pod sebou nebude mít žádné další položky, které by odděloval. Je zbytečné jej tam tedy mít. Princip nyní spočívá v tom, nastavit *visibility* na *FALSE* u položek, které nemají být zobrazeny a naopak na *TRUE* u položek, které zobrazeny být mají. Je to poměrně složitý algoritmus, který musí počítat s několika podmínkami:

1. Nadpis zabírá počet buněk, odpovídající počtu sloupců - chceme-li jej vykreslit, je potřeba nejprve skrýt další položky, které svou pozicí odpovídají řádku, kde má být

nadpis vykreslen.

2. Nadpis musí být na samostatném řádku - máme-li nastaveno zobrazení ve více sloupcích a počet jídel spadajících pod předchozí záložku není beze zbytku dělitelný tímto počtem sloupců, pak musí být zbývající tlačítka na posledním řádku skryta.

3. Na posledním řádku mohou být pouze položky, resp. tlačítka s názvy jídel.

4. Každá stránka musí začínat nadpisem - toto platí i pro nadpisy, nezačínající na aktuální stránce. Může tedy dojít k příkladu, který byl uveden u rozsahů. Máme 25 pizz a zobrazení na 10 položek na stránku. Datasource nám neumožní vygenerovat si duplicitní nadpisy. Teoreticky umožní, ale ne správně. Budeme např. mít před záložkou s 25 pizzami jinou záložku se dvěma položkami, která je momentálně vybrána. Za ní následuje záložka s pizzami. Máme-li zároveň zapnuté nadpisy, bude čítat datasource 29 prvků (1+2+1+25). První stránka zobrazí celou první záložku a z druhé záložky jen prvních 6 prvků. Druhá strana však musí začínat nadpisem *Pizzy*. Pokud bychom logicky na 10. pozici do *datasource* vložili duplicitní nadpis, vše by fungovalo výborně až do okamžiku, kdy by uživatel nejprve vybral záložku s pizzami, čímž by se na první stránce zobrazily nadpis *Pizzy* a výčet prvních 9 pizz. V tomto případě má *offset* již hodnotu 3 (včetně nadpisu) a jsou zobrazeny položky 3.-12. a nadpis na druhé straně již nepatří na 10. pozici nýbrž na 13.

   Takže tudy cesta nevede. Řekli jsme si, že *datasource* se plní jen při startu, takže ani načítat jej znovu po každém kliknutí na záložku také není ta správná volba. Nezbývá než při samotném vykreslování zjistit, jestli jsme na začátku stránky a ověřit jestli na toto místo přísluší v kolekci *datasource* nadpis. Pokud ne, tak je potřeba najít první předchozí a pro vykreslení další položky nezvyšovat *offset*.

   Problém s duplicitními nadpisy zasahuje také do vykreslování čísel stránek. Je tedy také potřeba při výpočtu počtu stránek započíst tyto položky do celkového součtu, což ve výsledku může zvýšit počet stránek.

4.3.2.5 Efektivita

Efektivita spolu s rychlostí je jedna z nejdůležitějších vlastností, na kterých ve výdeji záleží. Pro zlepšení efektivity můžeme využít některá vylepšení, která program nabízí.

Buben

Představme si, že máme v demonstračním příkladě 3 záložky a z toho každá obsahuje pouze 2 položky. V tomto okamžiku pracujeme pouze se základní verzí programu, jejíž chování zhruba odpovídá programu původnímu, popsanému v kapitole *současný stav*. Pro lepší obraznost budou ty záložky obsahovat *hlavní jídla*, *polévky* a *nápoje*. Nyní potřebujeme namarkovat od každého druhu jednu položku. Jednu polévku, jedno hlavní jídlo a jeden nápoj. Mezi jednotlivými výběry jídla však musíme vždy vybrat nejprve záložku, do které jídlo spadá a teprve pak můžeme vybrat další položku. Přitom se zobrazí vždy jen 2 položky (3 v případě zobrazování nadpisů) a zbytek obrazovky je prázdný a nevyužitý. První z vylepšení tedy umožňuje zobrazit položky postupně za sebou. V případě zobrazení nadpisů jsou položky od sebe odděleny právě nadpisem popisujícím záložku, do které následující položky spadají.

	Zákl. nastavení	Zapnutý buben	Zlepšení (v %)
Výběr 3 položek na jedné stránce	4.16	1.72	141%
Výběr 3 položek na ze 2 stránek	4.08	2.60	56%
Výběr 3 položek na ze 3 stránek	4.32	4.56	-5%

Tabulka 4.1: Zrychlení aktivací bubnu.

Pro ověření teorému přínosnosti bubnu jsme za pomoci pěti uživatelů získali časy, jejichž průměrné hodnoty ukazuje tabulka 4.1. Zde je vidět, jakého zvýšení efektivity jsme dosáhli. Vybírali jsme vždy 3 položky a každá z nich musela být v jiné záložce. V prvním případě se všechny záložky vešly na jednu stránku a zároveň došlo k nejlepšímu zlepšení a to až 141%. V druhém již byl výsledek horší a ne všechny vybírané položky byly na první stránce. Bylo tedy pro celý výběr potřeba vybrat další stránku. A ve třetím případě byly obsahy záložek tak veliké, že se nevešly na jednu stránku. Mezi každou vybranou položkou tak bylo pokaždé potřeba vykreslit obsah celé stránky znovu v obou případech.

Závěrem lze konstatovat, že buben přináší významné zlepšení. Převážně u záložek s malým počtem položek uvnitř. U záložek s velkým počtem položek sice nedochází ke zlepšení, ale ani se situace nezhoršuje.

Výhody této vlastnosti je možno ještě podpořit nastavením většího počtu sloupců a řádků. Tímto je možno docílit zrušení stránkování, ale hlavně získání zobrazení všech položek jídelníčku na jedné jediné stránce. Také se takto dá lehce přiblížit naměřenému zrychlení i při větším počtu položek.

Zvýšení počtu sloupců je programově omezeno na 5 sloupců. U řádků je tato mez mnohem výše. Ovšem v případě řádků dojde k omezení spíše než programovému, k omezení zrakovému (nápisy půjdou špatně přečíst) nebo k omezení způsobené nepřesností (na dotykových displejích nastanou problémy "trefování se" do malých tlačítek), proto není vhodné to s počtem moc přehánět.

Zjednodušené markování

Zjednodušit markování lze snadno u záložek, které mají uvnitř jen jednu položku. V takovém případě pak vždy, když je vybrána takováto záložka, lze vybrat pouze tuto jedinou položku. Dochází tak ke "zbytečnému" dvojkliku (nejprve na záložku, poté na položku).

	Základní nastavení	Zapnuté zj. mark.	Zlepšení (v %)
Více stránek s položkami	5.63	4.94	13%
1 stránka s položkami	4.87	5.02	-3%
Jednopol. záložky na začátku	5.09	4.94	3%

Tabulka 4.2: Zrychlení pomocí zjednodušeného markování.

Při měření byla k časům připočtena konstanta 4s, která podle předchozího měření odpovídá namarkování 3 položek. Jak ukazuje tabulka 4.2, pomůže tato schopnost zjednodušeného markování sice jen lehce a to přibližně 13%. A to jen za předpokladu, že jsou všechny položky zobrazeny na více stránkách a jednopoložková záložka je umístěna na konci (obecně na jiné stránce než je zobrazena). U následujících dvou měření nedošlo k žádnému zlepšení ani zhoršení. Rozdíl je jen odchylka odpovídající chybě měření.

Vlastnost má vužití jen pokud se nevejdou všechny položky na jednu stránku. Této vlastnosti se dá případně použít také pro vytvoření tzv. "speciálních tlačítek", které pomohou markovat nejčastěji odebírané položky. Takovými mohou být např. bagety, konkrétní druh nápoje, atp.

Rotování bubnu

Tato vlastnost, jak název napovídá, umožňuje rotovat buben automaticky. Ne zcela automaticky tak, že by se položky před očima měnily, to by opravdu nebylo moc přínosné. Automaticky se však dokáže otočit na další záložku, resp. její obsah. Namarkuje-li uživatel cokoliv ze záložky č.1, pak buben, v tomto okamžiku automaticky, odrotuje na obsah záložky č.2.

V tomto případě nemá moc smysl provádět měření. Zlepšení je v tomto případě silně závislé na konkrétním výběru jídla. Vyjdeme proto z naměřených hodnot předchozích měření. Opět budou markována 3 jídla.

V ideálním případě bychom vybírali položky ze záložek, které následují bezprostředně za sebou. Podle prvního měření bychom se tak měli dostat na hodnotu okolo 141%. To však platilo v případě, kdy byly zobrazené položky stálé. V tomto případě se budou měnit a dojde tak ke zpomalení, než oči stihnou nové texty přečíst. Zároveň však může dojít ke zrychlení. Budou-li obsahy záložek obsahovat jen malé množství položek, nebude potřeba téměř vůbec pohnout rukou, protože rotací budou všechny položky ze záložky stále pod rukou. V ideálním případě tak může dojít k ještě vyššímu zlepšení, než k jakému dojde jen samotným zapnutím bubnu. Může to být až ±150%.

Naopak, v nejhorším možném případě, budeme potřebovat vybrat všechny 3 položky ze stejné záložky. V takovém případě si práci spíše přidáme a budeme muset po výběru každé z položek navíc ještě znovu vybrat tutéž záložku. Místo 3 kliknutí budeme k úspěšnému namarkování muset kliknout celkem 5x. Zhoršení se pak vyhoupne na 66% plus režie, než oči zaostří po překreslení. Pokud by se markovalo položek více, došlo by k ještě vyššímu zhoršení.

	Zlepšení (v %)
Ideální případ	150%
Nejhorší případ	-70%

Tabulka 4.3: Zrychlení dosažené rotováním bubnu.

Někde mezi těmito dvěma extrémy se nachází reálná situace, která bude na každém provozu samozřejmě jiná. Někde se bude blížit spíše té první variantě, jinde zase té druhé. Tuto vlastnost nelze nikomu vnucovat a v některých případech by to bylo stejně jen k neprospěchu věci. Proto se nabízí možnost v nastavení tuto vlastnost úplně zakázat nebo povolit v závislosti na tom, jestli danému uživateli vyhovuje, či nikoliv.

Shrnutí

Na základě předchozích měření jsme zjistili, že všechna zlepšení vedou, za jistých okolností, ke zvýšení efektivity práce.

První z nich (buben) přináší zlepšení téměř vždy oproti původnímu modelu. Výsledná efektivita při jeho použití je však úměrná průměrné velikosti záložky. Čím menší je průměrný

počet položek uvnitř záložky, tím je dosahováno kratších časů a tedy i vyšší efektivity. Tato vlastnost nemá žádné vedlejší negativní účinky na jiné vlastnosti. Nezvratným faktem je, že toto vylepšení přináší v téměř 100% přínos.

Druhá vlastnost (zjednodušené markování) již nepřináší tak velké výhody, nýbrž závisí na méně pravděpodobné podmínce. Touto podmínkou je alespoň jedna záložka, která obsahuje pouze jednu položku, vhodně umístěnou nejlépe ve spodní části. Také předpokládá, že se celý výběr nevejde na jednu stránku. Tento stav nemusí sice nastat ani v jednom případě, ale aplikace je na něj připravena a dokáže uživatelům práci usnadnit.

Poslední vlastnost (rotace bubnu) dosahuje mnohem vyšších procent, co se týče zlepšení oproti předchozí vlastnosti. Jenže není všechno zlato, co se třpytí a tak zde dochází i k nepříznivému chování při potřebě markovat položky ze stejné záložky. Navíc také dochází k lehce matoucímu jevu, pokud je zároveň aktivováno zjednodušené markování. Při výběru záložky *Bageta*, která obsahuje jen jednu položku, dojde k okamžitému namarkování. Zároveň se rotací dostaneme na obsah další záložky např. *Nápoje*. Toto je sice logické chování vyplývající z nastavení, ovšem je docela matoucí, pokud uživatel klikne na záložku bageta a místo baget se zobrazí nápoje.

Složení jídelníčků se různí, stejně jako jsou různé i jednotlivé provozy. Nelze tedy dopředu nastavit, jak se bude program na jednotlivých provozech chovat. Lepší variantou je nechat uživatele, aby si sami nastavili, co jim právě v daný okamžik lépe vyhovuje. Dokonce ani buben nemusí být přínosný, pokud je uživatel již zvyklý na původní schéma a toto nové jej mate a znepokojuje. Všechny vlastnosti jsou k dispozici v nastavení a jsou vztaženy na konkrétní uživatele. Jejich nastavení tak neovlivňuje preference jiných uživatelů. Možností pružně měnit chování programu za chodu je možné dosáhnout té nejvyšší možné efektivity.

4.3.2.6 Uživatelská přívětivost

Sebevíc kvalitní, výkonná a rychlá aplikace může narazit na nezájem ze strany uživatelů velice snadno. Může totiž postrádat uživatelskou přívětivost. Ani na tento aspekt nebylo při tvorbě programu zapomenuto. Dokonce byl brán v úvahu jako jeden z těch nejdůležitějších, hned po rychlosti, bez které se aplikace neobejde v žádném případě.

Ne vždy se vývojář může zavděčit všem uživatelům. Jeden bude chtít využívat jeden sloupec na který je zvyklý, jiný bude chtít zase využít maximálně plochu, která se mu nabízí, aby se vyhnul použití stránek. Další má rád modrou barvu, proto se mu nebude líbit barevné řešení, které aplikace nabízí. Jak již bylo zmíněno výše, někomu se nemusí zamlouvat zapnutý buben atp.

Se snahou zavděčit se většině uživatelů a zodpovědět tak jejich dotazy, proč je tohle takové jaké to je a ne takové, jak by se mi to líbilo více, vzniklo velice pružné nastavení. Nastavení, které za běhu programu může pomoci upravit program tak, aby odpovídal co nejvíce představám samotných uživatelů.

Volby

Nastavení umožňuje, kromě změny chování, také nastavení adekvátního vykreslování. Ať už jde o možnost zobrazení nebo nezobrazení nadpisů, počty sloupců, počty zobrazených číslic stránek nebo o počty řádků. Je tím tak snadno vyřešena situace, kdy jednomu uživateli přebývají na druhé stránce 2 položky, ale kvůli jiným uživatelům, kteří mají problémy se zrakem, není možno v programu počet řádků zvyšovat.

Výčet položek, které lze uživatelsky za běhu programu nastavovat následuje:

- Zapnout/vypnout buben
- Zapnout/vypnout rotaci bubnu
- Zapnout/vypnout zjednodušené markování
- Zapnout/vypnout zobrazení nadpisů
- Zapnout/vypnout zobrazení záložek
- Zapnout/vypnout zobrazení čísel stránek
- Změnit počet řádků v rozsahu 2-40
- Změnit počet sloupců v rozsahu 1-5
- Změnit počet čísel stránek, která se zobrazí najednou

V nastavení se nově mohou také "vyřádit" uživatelé, kteří mají potřebu vlastního barevného řešení. Změnit si mohou jak barvy položek, tak i barvy záložek. Pro úplnost lze změnit i barvu pozadí.

Při výběru barvy tlačítek mají uživatelé naprostou volnost. To znamená, že si mohou vybrat i velmi tmavé barvy, na kterých pak nebude vidět nápis s názvem. I toto je řádně ošetřeno. Nastane-li tato situace, pak se zcela automaticky změní barva písma na barvu bílou.

Nesmíme také zapomenout na uživatele se silným vztahem k rodině. Nebo obecně k těm, kteří mají něco velmi rádi a rádi to mají na očích. Většina z nás nosí fotky svých milých v peněženkách nebo v dokladech, někteří mají fotku rodiny nebo někoho blízkého i na pracovním stole. Pro takové osoby, které jsou "odsouzeny" sedět na pokladně celou pracovní dobu, se nabízí možnost vložit si vlastní obrázek na pozadí celého výdejního formuláře.

Jak již bylo naznačeno výše, změny se vztahují přímo na konkrétní uživatele, nikoliv na celý systém nebo konkrétní provoz. Díky tomuto vztahu je dokonce možné jít tak daleko a nabídnout "vlastní" nastavení svému uživateli i pokud změní terminál. Dnes sedí u terminálu 1, zítra bude u 2 a určitě velice potěší, když si své oblíbené nastavení nebude muset zadávat znovu.

Barvu lze měnit u těchto položek:

- První barva záložky
- Druhá barva záložky
- Barva pro ukončení menu
- První barva položky
- Druhá barva položky
- Barva pozadí
- + Nastavení obrázku na pozadí

U položek a záložek jsou vždy dvě barvy. Jak bylo v kapitole o přehlednosti v sekci barvy, je pro zvýšení přehlednosti důležité, aby se barvy tlačítek střídaly a nesplývaly tak před očima dohromady.

Nápověda

Poslední z věcí pro podporu uživatelů, která zde také patří, je existence nápovědy. Důležitost nápověd je značná, převážně u nového programu se kterým uživatelé nejsou zvyklí pracovat.

Nápovědy jsou dostupné z každého dialogu. Nejen z výdeje stravy. Jsou psány v HTML kódu a následně převedeny do standardní windowsovské nápovědy. Každá z nich obsahuje obrázek se screenshotem příslušného okna a popis, co daný dialog dělá a jak se s ním pracuje.

4.3.3 Vydávané položky

Vztah mezi vydávanými položkami a celým jídelníčkem již byl vysvětlen v jedné z předchozích kapitol. Nicméně pojďme se podívat na volbu, která rozhodne o tom, co by mělo být zobrazováno při výdeji stravy na pokladnách.

První logicky se nabízející možností jsou právě *vydávané položky*. Vzhledem k tomu, že se jedná o podmnožinu z celého denního jídelníčku, tak získáme menší počet položek, které bude následně potřeba zobrazit.

Druhou variantou je zobrazit celý jídelníček. Ten, na rozdíl od první varianty, zabírá více místa. Zato má jednu obrovskou výhodu a tou je neměnnost.

Studie práce na stávajícím systému však ukázaly, že uživatelé mnohdy ani nečtou nápisy na jednotlivých tlačítkách, ale markují jídla tzv. poslepu. Jsou zvyklí, že tohle jídlo je na 5. od vrchu a tamto zase úplně dole. Pokud bychom jim měnili uspořádání a složení jednotlivých záložek tzv. "pod rukama" nebyli by z toho určitě moc nadšeni. Bohužel by došlo i ke zvýšení chybovosti markování jednotlivých odběrů.

Další skutečností, která "hraje" pro zobrazení celého jídelníčku, je ta, že by tak mohlo docházet k následujícím situacím. Kuchařky vydají poslední jídlo daného druhu a odstraní jej tedy z vydávaných položek. Pokud by se strávník, který si toto jídlo vybere, zdržel např. výběrem nápoje, tak ve chvíli, kdy by došel ke kase, už by jej nebylo možno namarkovat, protože už by nebylo ani v nabídce.

4.3.4 Offline provoz

Jakkoliv může být aplikace ošetřena a vypilována do posledního detailu, může dojít k nečekanému problému. Chceme-li zaručit programu spolehlivost, budeme muset myslet dopředu a být připraveni na možné problémy, které mohou v plném provozu nastat. Nemusí to však být zaviněním aplikace, nýbrž problém uvnitř aplikace může vyvolat nečekaná situace, která nastane vně a se kterou nepočítáme.

Takovou situaci může zapříčinit například výpadek proudu, který samozřejmě nezachrání kvalitní program, ale technická zařízení jakou je UPS atp. Jejich použití nesouvisí s aplikací. Co s ní ovšem souvisí je např. řešení problémů v případě výpadku serveru nebo při problému s připojením.

Tuto situaci řeší tzv. *Offline provoz*. Tento režim by měl umožnit pracovat i bez nutnosti komunikace se serverem. Obecným předpokladem je skutečnost, že se jídelníčky plní přibližně týden dopředu. Proto není problém, aby na každém terminálu byl jídelníček připraven na několik dní dopředu.

Nelze však zabránit skutečnosti, že další den provozu bez připojení již nemusí být některé údaje aktuální. Tento problém bude nejvíce hrozit u cen, které se budou měnit v závislosti na skladových cenách. S tímto nedostatkem je tedy potřeba počítat a nelze očekávat, že by jej program řešil.

Ještě větším problémem jsou stavy kont jednotlivých strávníků. Pokud nedojde k offline režimu, pak se při každém odběru zjistí aktuální stav konta strávníka. V případě offline režimu k této komunikaci bohužel nedojde a tak se musí použít poslední známá hodnota z doby, kdy ještě spojení se serverem fungovalo.

Tento stav je v současné době v původním systému vyřešen papírem a tužkou. Jinak řečeno, vyřešen není a uživatelům tak nezbývá, než zapisovat si jednotlivé odběry na papír. Při zprovoznění serveru pak musejí tyto odběry dodatečně do programu zadávat.

Této situaci se pokusíme předejít právě oním stahováním jídelníčků, příslušných katalogů, cen a údajů o kontech na lokální terminály. V případě výpadku, po několika neúspěšných pokusech navázat spojení se serverem, systém přejde do offline režimu a uživatelé budou moci beze změn dále pracovat. Offline režim podporuje jen a pouze výdej stravy. Není v tomto režimu tedy možné upravovat jídelníčky, ceny ani katalogy.

Na rozdíl od serveru, kde pracuje výkonný MS SQL server, je na lokálních stanicích pouze MS Access. Tato databáze se chová v některých případech trochu jinak. Jedním z rozdílů je spouštění více SQL dotazů v jednom. Také jsou zde jiné funkce. Převážně jsou potřebné pro práci s daty a i ty jsou jiné.

Proto musí mít každá z příslušných tříd, které jsou pro offline provoz nezbytné, dvě různé implementace metod, které na databázové vrstvě programu obstarávají načítání objektů.

Tyto metody budou navíc ještě potřebovat vlastní tabulky v lokální databázi. Vytvořeny budou:

- VY_Jidelnicky
- VY_DefiniceMenu
- VYC_DobyVydeje
- VYP_ZalozkyTSKomodit
- KAC_Kategorie
- KAC_Komodity
- VY_Odbery
- VY_SlozeneVydeje

Kromě těchto několika málo tabulek je v lokální databázi několik dalších, které se také starají o bezproblémový chod v případě výpadku.

Obsah těchto tabulek nejsou přesnými kopiemi odpovídajících tabulek na serveru. Vypuštěny byly totiž některé sloupce, jako je např. IDgProvozu atp. ID provozu není na lokální stanici potřebné. Tam víme o jaký provoz se jedná a je také zbytečné zde mít jídelníčky provozů jiných. Dalším příkladem "odlehčení" tabulek je například ID kategorie u samotných komodit. Také je zbytečné uchovávat informace o komoditách, které se na tomto provozu nevydávají, atp. Kopírování na lokální stanici sice netrvá extrémně dlouhou dobu, ale snahou je i tento čas co nejvíce zkrátit. Proto jsou na lokálních stanicích jen nezbytně nutná data. Tímto způsobem dochází také k ulehčení vyhledávání a práce s Accessovými databázemi v případě potřeby. Tyto databáze nejsou stavěny pro velké objemy dat, tak jako je tomu u MS SQL serveru.

Pokud by se data "přelévala" do lokální databáze pouze při startu aplikace, pak by se toto také dalo považovat za řešení. Pokud se však oprávnění uživatelé rozhodnou změnit

např. prodejní vzorce během dne, pak by všichni ostatní uživatelé, pracující s výdejem stravy, museli zavřít aplikaci a spustit ji znovu. O této změně by se museli ale nějakým způsobem dozvědět.

O tento problém se nám postará synchronizační objekt, který v pravidelných intevalech porovná vybrané tabulky a podle specifikovaného SQL příkazu zjistí, jestli došlo ke změně v datech, týkajících se právě tohoto provozu a dat k němu relevantních.

Problém je takto elegantně vyřešen a navíc v případě výpadku spojení se serverem máme v lokální databázi data stará několik minut a ne ta, která byla platná při prvním spuštění.

4.3.5 Udržování aktuálního jídelníčku

Cena jako taková je závislá na několika faktorech (např. typ načtené karty, omezení, atp.) a nemůže být proto vázána na jednotlivá tlačítka tak, jako jsou vázány položky jídelníčku. Proto se načítá vždy znovu podle přiložené čipové karty. Pokud dojde k aktualizaci cen, tak hned při dalším odběru jsou ceny opět aktuální. Problém ovšem nastává, pokud se změní cokoliv ze zobrazeného výdeje stravy.

Je nesmyslné načítat znovu všechny položky při každém novém odběru. To bychom se dostali opět na začátek k současnému stavu. Již jsme si uvedli, že tato situace není ideální.

Řešení tohoto problému se nabízí hned několik.

1. Základní varianta, která je pro vývojáře bez nutnosti jakékoliv aktivity, je nechat uživatele, aby si tuto situaci vyřešili sami mezi sebou a oznámili si, že došlo ke změně. Jenže jak poznají, kdy jim o změně mají říct? Mohou provést změnu v názvu komodity a pak porovnat podle jídelníčků, kde se to nyní vydává a těm zavolat. Toto rozhodně není přijatelné řešení.

2. Druhou variantou by bylo řešení, kdy se na serveru pro každý provoz vytvoří příznak. Tento příznak by, v případě jeho nastavení, mohl nést informaci o neaktuálním zobrazení na vybraném terminálu. Nevýhodou je, že takovýto přístup nepozná, jestli je na daném terminálu načten jídelníček pro obědy, večeře, či snídaně. Pokud objeví příznak, překreslí se nezávisle na skutečnosti, jestli opravdu bylo zobrazení neaktuální.

3. Poslední nabízenou variantou je uložení informací o tom, které položky jsou právě zobrazovány. Uložení by proběhlo přímo do lokální databáze do speciálních tabulek. Tento přístup bere v úvahu i dobu výdeje, která je právě načtena.

Snahou je samořejmě vytvořit program co nejvyšší kvality a proto byla také zvolena poslední varianta. Sice je nejvíce pracná, ale její chování nejvíce odpovídá požadovanému cíli.

Pro uložení informace o aktuálně zobrazeném jídelníčku poslouží v lokální databázi tyto tabulky:

- VY_JidelnickyZobrazeno
- VY_DefiniceMenuZobrazeno
- KAC_KomodityZobrazeno
- VYP_ZalozkyTSKomoditZobrazeno
- KAC_KategorieZobrazeno

Jak je vidět, nebudou se kontrolovat jen složení jídelníčků, ale i konkrétní jídla a záložky. Pod kontrolou tak budou i změny v názvech nebo v pořadí záložek, atp.

Samotné uložení je velice jednoduché. Ve chvíli, kdy se načítá, co se má zobrazit, jsou vždy v lokální databázi aktuální data a jsou tedy shodná s těmi, která se zobrazí. Uložíme-li je ve stejný okamžik do připravených tabulek, budou odpovídat těm zobrazeným. Aby bylo uloženo jen to, co je zobrazeno, vybere se jídelníček s dnešním datem a s vybranou dobou výdeje. Po získání jídelníčku se naplní i ostatní tabulky. Plnit se musí v přesně daném pořadí, ve kterém jsou uvedeny výše. Důvodem je závislost hodnot, které do tabulky patří, na tabulce nebo tabulkách, které jsou nad ní. Např. tabulku zobrazených komodit naplníme jen komoditami, které jsou obsaženy alespoň v jedné z předchozích tabulek, tedy v jídelníčcích a definicích menu. Komodity, jejichž ID není v jedné z předchozích tabulek, nás momentálně nezajímá. Vlastně ano, ale zde patří jen data o zobrazených položkách.

Máme tedy uloženy všechny informace o tom, co je zobrazeno v tabulkách s přívlastkem "zobrazeno". Také máme v normálních tabulkách načteny aktuální hodnoty, které sem pravidelně dostává synchronizační objekt, zmíněný již dříve. Co však nyní potřebujeme, je způsob, jak efektivně porovnat, jestli nedošlo k nějaké změně, kvůli které by bylo potřeba znovu překreslit výdej stravy.

V ideálním případě by tento způsob nemusel porovnávat každou tabulku zvlášť, aby se ušetřil čas při komunikaci s databází. Velice efektivní je SQL dotaz, založený na tomto jednoduchém principu, který lze demonstrovat na tabulkách jídelníčků:

Z obou dvou tabulek (myšleny jsou dvojice s a bez přívlastku zobrazeno) se vyberou adekvátní položky. V případě tabulky zobrazeno to je celá tabulka. V druhém případě tabulka s omezením na den a dobu výdeje. Oba výsledky se spojí a klauzulí GROUP BY se spojí podle všech sloupců. Druhá klauzule

```
HAVING COUNT(*)=1
```

zaručí, že budou vybrány jen ty položky, které se neshodují ve všech slopcích s nějakou položkou v druhé tabulce.

Výsledek je buď

1. žádný záznam - tabulky jsou tedy identické
2. jeden záznam - pokud byla položka přidána nebo smazána
3. dva záznamy - pokud byla položka změněna
4. tři a více záznamů - kombinace několika předchozích možností

Takto jsme zpracovali jednu tabulku. Podobně lze zpracovat i ostatní tabulky a mezi jednotlivé výsledky stačí vložit UNION ALL. Takto získáme počet záznamů ze všech tabulek jedním dotazem. Je-li výsledek 0 záznamů, pokračujeme dále v práci. V opačném případě vyšleme informaci zobrazovacímu modulu, aby provedl znovunačtení. K zobrazení aktuálních hodnot ovšem nemůže docházet během započatého odběru. Je nutno počkat až na zakončení stávajícího odběru. Pokud ve chvíli, kdy byla neaktuálnost zjištěna a neprobíhá žádný odběr, pak se aplikace sama překreslí. V obou případech obsluhu upozorní na změnu.

4.3.6 Stornování

Storno je ve výdeji považováno za zrušení nějaké části odběru.

Občas při výdeji totiž dochází k tomu, že pokladní omylem namarkuje jinou položku, než jakou si strávník vybral. V tomto případě by mělo dojít ke stornování tohoto odběru nebo jeho části a namarkování správných položek.

Pokud k této situaci nyní dojde, nemá pokladní možnost vzniklou situaci (ve stávajícím systému) nijak napravit. Jediným východiskem je cesta strávníka na vedení, kde tento odběr může pověřená osoba stornovat. Toto řešení není ekonomické ani pro strávníka ani pro osobu, která s ním tento problém musí řešit.

Stornovaní by mělo být přístupné nejen v příslušném dialogu, který je pro práci s odběry určený a dokáže řešit otázku stornování, ale také přímo v dialogu výdeje stravy.

Řešení je poměrně jednoduché. V dialogu výdeje stravy lze delším přidržením příslušného tlačítka možné vyvolat režim stornování. Po tomto lze nechat strávníka, aby přiložil kartu a tím se zobrazí celý poslední nestornovaný odběr. Lze pak vybrat položky, které mají být stornovány a potvrdit volbu stejně jako u běžného markování.

Implementační část pro zobrazení jednoduše zobrazí odběr s nestornovanými položkami, tedy těmi, které nemají příznak storno. Problematičtější je ovšem druhá stránka věci a tou je *kniha plateb*. Každý provedený odběr je zapsán do *knihy plateb*, kde musí být všechny záznamy sekvenčně očíslovány. Není možné odstranit nějakou položku, pokud již za ní existuje jiná s vyšším číslem. Proto ani storno není tak snadnou operací. Nelze jednoduše odběr smazat. Jedinou možností je vytvořit tzv. "protikus", který do knihy plateb vloží zápornou hodnotu a její opak se pak připíše na konto strávníka.

4.4 Lokalizace

Stravovací informační systém je komerční systém a jako takový se pochopitelně snaží získat co největší uplatnění. Jednou z vlastností, která tomuto napomáhá je i lokalizace.

Lokalizace je ve své podstatě překlad jazykových částí programu. Pro snadný překlad a změnu jazyka, kterým s uživateli komunikuje, je potřeba, aby veškeré textové řetězce byly umístěny mimo samotný program.

Tuto vlastnost má i tento stravovací systém a taktéž i tato výdejní část. Všechny řetězce jsou zapsány do excelovské tabulky, kde má každý řádek přiřazeno sekvenční číslo a jednotlivé sloupce pak zobrazují texty v různých jazycích.

Program samotný však neumí pracovat s excelovskými tabulkami a proto je potřeba výstup upravit příslušným makrem do datového souboru. V programu jsou všechny číselné řetězce, které jsou přímo na formulářích (např. jako popisky(label)), převáděny na text uvedený v převodním souboru. Ostatní texty jsou pomocí statické funkce GetResString(číslo) a nastaveného jazyka převáděny na správný text.

Při implementaci bylo hojně využíváno nápovědy umístěné zde: [3] a také návodů a poučných článků umístěných zde: [2]

Kapitola 5

Závěr

Prvním z úkolů této práce bylo seznámení se s technologií .NET.

Po důkladném seznámení, které probíhalo poměrně dlouhou dobu (a stále ještě trvá, protože objevuji nové možnosti), musím podotknout, že pro vývoj informačního systému je tato technologie zcela ideální. .Net technologie je obrovským přínosem nejen pro programátory informačních systémů.

Kromě plně objektového přístupu a kompatibilitě různých programovacích jazyků, také velkou měrou dopomáhá velmi kvalitně propracované Visual Studio. Ať již díky intellisence (napovídání), které je ve Visual Studiu výborným pomocníkem při tvorbě aplikací, nebo propracovaným prostředím pro tvorbu uživatelského rozhraní.

Práce se zabývá celkovým vývojovým cyklem informačního systému. Od počáteční analýzy, přes návrh až k samotné implementaci. Dle požadavků byl návrh proveden doslova na míru pro menzy provozované KaM VUT Brno. Důraz byl však také kladen na to, aby byl program dostatečně univerzální a aby jej tak bylo možno využít i pro jiné menzy. Lokalizace pak otvírá další možnosti využití programu i pro zahraniční stravovací provozy.

Ponevadž se jedná o komerční aplikaci, musí být všechny chyby důkladně ošetřeny a jejich případné výskty logovány.

Za hlavní přínos práce považuji řešení problematiky offline provozu. Mé řešení offline provozu spočívá ve využití jedné duplicitní lokální databáze, která vedle té hlavní serverové, umožní v případě výpadku, fungování aplikace i po přerušení síťového připojení, což je klíčovou vlastností systému z provozního hlediska.

Po získání spojení se vzdálenou databází je poté potřeba správně sesynchronizovat všechny akce, které byly během výpadku provedeny.

Kvůli možnému výpadku je potřeba pravidelně a efektivně spouště synchronizaci a synchronizovat obsahy vzálené i lokální databáze. Synchronizace však musí probíhat tak, aby nedocházelo ke zpomalení markování odběrů, které mohou probíhat na popředí.

Předností tohoto řešení je to, že obsluha nemusí ručně spouštět synchronizaci a na lokální stanici je stále aktuální stav všech kont, jídelníčku atp. Přičemž při samotném provozu není zpomalení (způsobené stahováním) nijak poznat. Dojde-li pak k výpadku, aplikace může i nadále pracovat bez jakýchkoliv problémů.

Dalším přínosem je také nové zpracování uživatelského rozhraní pro dotykové displeje. Velký důraz byl kladen na maximální přispůsobivost uživatelské rozhraní a kromě nastavení barev, které zcela jistě využijí lidé, kteří mají lepší cit pro barvy než já, bylo důkladně propracováno nastavení zobrazení a chování. Uživatelé si tak mohou na každém provozu zvlášť nastavit způsob zobrazení a chování aplikace podle konkrétní situace na daném provozu. Výrazně také bylo zrychleno vykreslování.

Nyní se program nachází v testovací verzi a je testován zaměstnanci KaM VUT Brno. V nejbližší době je očekávána zpětná vazba, která pomůže program doladit, díky jejich dlouholetých zkušeností s výdejem stravy. Do ostrého provozu přejde aplikace již 1.7.2007.

Jako hlavní přínos považují kromě získání značného množství zkušeností s vývojem aplikací na platformě .NET ,také získání praxe při tvorbě komečního informačního systému. Další dobrou zkušeností je vytvoření programu od úplného začátku počínaje analýzou a návrhem a následným dotažením programu do konce až k ostrému provozu.

Budoucí vývoj by se měl hlavně zaměřit na zapracování návrhů získaných ze zpětné vazby. Vzhledem k tomu, že byl výdejní dialog navržen tak, aby jej mohli efektivně používat hlavně praváci, bude potřeba také zvážit úpravy i pro leváky.

Literatura

[1] doc.Ing.Jaroslav Zendulka,CSc., Ing. Vladimír Bártík, Ing. Šárka Květoňová. *Stujní opora k předmětu AIS*. 2006.

[2] WWW stránky. Diskusní fóra, seriály, články a informace o jazyce visual basic .net. `http://www.visualbasic.cz/`.

[3] WWW stránky. Msdn - microsoft development network. `http://msdn2.microsoft.com/en-us/default.aspx`.

[4] WWW stránky. Wikipedia - informační systémy. `http://cs.wikipedia.org/wiki/Informa%C4%8Dn%C3%AD_syst%C3%A9m`.

[5] WWW stránky. Wikipedia - .net. `http://cs.wikipedia.org/wiki/.NET`.

Dodatek A

Zadávací dokumentace

A.1 Úvod

Tento dokument obsahuje zadávací dokumentaci k modulu Stravování informačního systému dodávaného firmou ApS Brno s.r.o.. Systém je určen primárně pro evidenci výdeje stravy, k tomu ovšem nutně patří další činnosti, jako je doplňkový prodej, systém objednávek, evidence identifikačních průkazů - všechny tyto činnosti modul zajišťuje buď sám, nebo využívá služeb jiných částí informačního systému.

Systém má několik částí, kterými se zabývají následující kapitoly. Kapitoly jsou poměrně nezáživné na čtení - jde o seznam činností, které systém bude nebo naopak nebude umožňovat a seznam vlastností, které bude mít.

Jednotlivé kapitoly jsou provázané a o některých vlastnostech systému se hovoří na více místech. Pokud někde dochází k zdánlivému nesouladu textu - např. věta strávníkem může být kdokoli je v rozporu s omezeními vyplývajícími z blokace karty - platí vždy ta nejvíce omezující podmínka (zde tedy ta o blokaci karty). Rozpory typu "strávník vždy smí to" a "strávník nikdy nesmí to" se doufám nevyskytují.

A.1.1 Obsah kapitol

- Následující kapitola, *Architektura systému*, stručně shrnuje jak systém vypadá a jak je vnitřně propojen a jaké technologie používá a vyžaduje.
- Kapitola *Terminály* dělí zařízení, která se v systému mohou vyskytovat, podle jejich funkce.
- Kapitola *Strávníci* popisuje vztah osob k systému a způsoby jejich identifikace. Dále uvádí údaje, které se o strávnících evidují a shrnuje strukturu uchovávaných údajů.
- V kapitole *Strava* se popisuje co a jak systém vydává a také popisuje zpracování jídelníčků.
- Kapitola *Ceny* uvádí způsoby zjištění ceny stravy - systém určuje cenu podle strávníka a odebírané stravy. Rovněž zde najdeme popis chování systému v limitním a bezlimitním způsobu určování ceny.
- Kapitola *Výdej* shrnuje požadavky na výdejní část systému - co musí systém zajistit, co naopak nesmí dopustit a co je v kompetenci obsluhy. Specifikuje požadavky na objednávkový a bezobjednávkový provoz a na provoz kombinovaný.

- Další kapitoly - *Správa* a *Vstupy a výstupy* - popisují možnosti správce systému a rozhraní pro vstup a výstup dat.
- Z Poslední kapitoly - *Evidované údaje* - vyplývá, jaké informace bude systém o svých jednotlivých částech uchovávat, uvádět tyto seznamy do textu by bylo na úkor přehlednosti.

A.2 Architektura systému

- Z vnějšího pohledu má systém 2 vrstvy - klientskou a serverovou
- Z vnitřního pohledu se jedná o třívrstvou aplikaci - aplikační logika je součástí programu na straně klienta, stejně jako uživatelské rozhraní. Datová vrstva leží na serveru.
- Server (databázový) bude MSSQL Server ve verzi 2000 či 2005.
- Klientské pracoviště bude PC nebo jiné zařízení. Pracoviště bude schopno provozovat aplikaci pod operačním systémem Windows 2000 nebo kompatibilním a pracovat s daty na databázovém serveru.
- Pracoviště může být vybaveno různými dalšími periferiemi (pro seznam základních vizte kapitolu Správa - umožňuje veškeré činnosti systému včetně funkcí uvedených v kapitole Správa
- Podporovaná periferní zařízení), systém s nimi bude umět pracovat za předpokladu, že bude dostupná specifikace komunikačního protokolu zařízení a (budou-li potřeba) ovladače tohoto zařízení pro systém Windows 2000 (nebo kompatibilní).
- Některé části systému budou realizovány s využitím webového prohlížeče jako tenkého klienta a webového serveru jako střední vrstvy. Půjde zejména o systém zadávání a sledování objednávek stravy.
- Připojení k serveru bude záležitost standardních metod operačního systému. Pro zabezpečenou komunikaci s databázovým serverem bude třeba využít VPN, zabezpečení komunikace mezi klientem a serverem nebude informační systém řešit.
- Klient bude schopen krátkodobě pracovat bez připojení k serveru - jako lokální úložiště dat bude použit MSDE nebo SQL Express server (bezplatná verze MS SQL 2000 či 2005).
- Systém bude pracovat s více databázemi, potenciálně umístěných na různých serverech. Modul stravování bude využívat tyto databáze - jednu systémovou a jednu vlastní, navíc bude nepřímo využívat databáze spolupracujících modulů.
- Systém bude pracovat s MS SQL 2000 i 2005 současně - různé databáze mohou být umístěny na různých verzích serveru.
- Systém umožní vstup a výstup dat. Vstupem se myslí načtení XML souboru daného formátu do systému a čtení dat z databáze. Výstupem se myslí současně tisk na tiskárnu, export do XML souboru daného formátu i zápis do databáze. Konverze do dalších formátů je možná jako rozšíření systému. Konkrétní seznam vstupů a výstupů obsahuje kapitola Vstupy a výstupy.

- Systém bude připraven na práci s digitálními podpisy a digitálními značkami (dle zákona č. 227/2000 Sb.) u výstupů do XML a digitálních exportů tiskových sestav. Systém bude připraven poskytnout nástroje pro ověření podpisů/značek.

A.3 Strávníci

- Strávník je označení toho uživatele systému, který odebírá stravu. Strávníkem může být více osob (pokud používají společně kartu/y).
- Strávníkem může být kdokoli.
- Strávník má v systému právě jedno konto. Není možné mít společné konto pro více strávníků, je ale možné rozlišovat odběry podle jednotlivých karet (viz kapitoly Ceny, Výdej a Vstupy a výstupy).
- Systém umožní pracovat s anonymními strávníky - strávníky, kteří mají kartu, ale nejsou známy jejich osobní údaje
- Systém neřeší problematiku zákona č. 101/2000 Sb. nad rámec toho, že zpřístupní osobní informace v systému právě osobám oprávněným se systémem pracovat
- Systém podporuje hotovostní strávníky - strávníky, kteří nemají žádnou kartu
- Strávníky eviduje společná část systému, je tedy společný s ostatními moduly (např. s Ubytováním)

A.3.1 Karty

- Karta je libovolný prostředek, kterým se strávník identifikuje (nejčastěji čipová karta).
- Typy karet (z hlediska použité technologie) shrnuje kapitola Způsoby identifikace strávníků
- Strávník může mít více karet
- Platnost karty může být časově omezena
- Systém musí umožnit zablokování a odblokování karty.
- Pro usnadnění některých operací (zejména tvorby vyúčtování) dělí systém karty do skupin, zpravidla podle příslušnosti k odběrateli.
- Platnost skupiny karet lze omezit na část organizační struktury systému (provoz či nákladové středisko).
- Systém bude evidovat výdej karet
- Systém umožní pravidelnou aktualizaci databáze karet (nebo její části) z externího zdroje (bude-li tento přístupný a zdokumentovaný).

A.3.2 Způsoby identifikace strávníků

- čipová karta
- průkaz s čárovým kódem
- stravenka s čárovým kódem (nepřímá identifikace)
- bez identifikace ("cizí" strávník)
- ověření třetí stranou (při přístupu přes virtuální terminál)
- systém umožňuje použít i jiné ověření strávníka (biometrický údaj, ...). Podmínkou je funkční připojení čtečky k pracovišti (COM, USB+ovladače) a specifikace komunikačního protokolu.

A.3.3 Objednávky a podobjednávky

- Informaci o podmínkách výdeje stravy váže systém na objednávky a podobjednávky
- Objednávka má jednu a více podobjednávek
- Podobjednávka určuje ceník, příspěvky na stravu a omezení výdeje

A.4 Terminály

- Terminál - zařízení na místě, kde dochází k interakci strávníka a systému. Zařízení komunikuje se systémem a zobrazuje odpovídající informace z něj a/nebo informace do systému vkládá, upravuje je v něm a/nebo maže. Terminál může provádět jen některé z uvedených aktivit (komunikuje vždy).
- Terminál má jednu nebo více z těchto základních funkcí:
  - Pokladna - umožňuje strávníkovi složit peníze na konto a čerpat z něj. K tomu je schopna identifikovat strávníka či objednávku stravy (stravenku).
  - Výdejní místo - slouží ke kontrole, zda je možno stravu strávníkovi vydat (např. zda strávník odebírá stravu, kterou si objednal) a k evidenci výdeje. Umožňuje identifikovat strávníka i objednávku stravy (stravenku).
  - Objednávkový terminál strávníka - umožňuje zadávat a rušit vlastní objednávky, používat burzu stravenek. Umožňuje tisk stravenek.
  - Objednávkový terminál obsluhy - umožňuje zadávat, prohlížet a vyúčtovat objednávky strávníkům, kteří se u terminálu identifikují . Umožňuje tisk stravenek.
  - Prohlížení jídelníčku - zobrazuje jídelníček na aktuální den, umožňuje-li to hardware pak i na jiné, uživatelem vybrané dny .
  - Správa konta strávníka - umožňuje strávníkovi zobrazovat stav konta a přehled odběrů za zvolené období.
  - Virtuální terminál - webová stránka s funkcemi objednávkového terminálu strávníka, prohlížení jídelníčku a správu konta strávníka.

  - Správa - umožňuje veškeré činnosti systému včetně funkcí uvedených v kapitole Správa A. Podporovaná periferní zařízení

- Systém bude určitě pracovat s těmito zařízeními:
  - Čtečka čipových karet
  - Čtečka čárového kódu
  - Terminál platební karty
  - Elektronická váha
  - Elektronická pokladna (displej, šuplík, tiskárna)
  - Tiskárna dokladů

A.5 Strava

- Strava - jakákoli položka, jejíž tok (výdej) bude systém sledovat. Nejedná se nutně o potravinu
- Systém bude sledovat výdej stravy - kdy, kde, komu a za jakou cenu byla strava vydána.
- Systém bude evidovat výdej stravy v běžných jednotkách - v kusech (nejčastěji), v litrech a v kilogramech
- Ve spojení s elektronickou váhou (ale i bez spojení) bude systém schopen vydávat po částech měrné jednotky (např. uzeniny či saláty)
- Systém umožní vést jako jednu položku výdeje i více fyzických věcí současně (např. menu - polévka, hlavní jídlo a nápoj)
- Systém bude odlišovat různé skupiny stravy - zejména pro účely zpřehlednění výdeje.
- Skupiny stravy budou mít hierarchickou (víceúrovňovou) strukturu . A. Jídelníčky
- Systém bude umožňovat tvorbu a správu jídelníčků
- Jídelníček bude obsahovat položky výdeje s časovou platností doby vydávání či bez ní (např. den, týden, stále). Časová platnost položky se uvádí ve dnech den.
- Jídelníčku bude možno přiřadit kategorii, seznam kategorií bude hierarchický a bude možno jej rozšiřovat a upravovat. Bude rovněž možno používat kategorie z modulu Sklad.
- Jídelníček se bude vztahovat na jeden provoz. Pro jiné provozy půjde zkopírovat.
- Položku jídelníčku bude možno zařadit jako tzv. společné jídlo k jiné položce - společná jídla bude možno vydávat pouze s touto (jinou) položkou. Příklad - polévku je možno vydat pouze s hlavním jídlem.
- Jídelníček bude možno vytvářet ve více jazycích, u zobrazení bude možno vybrat jazyk či zobrazovat více jazyků současně. Jeden jazyk je hlavní, systém podporuje zadat přes dvě stě jazyků jídelníčku.

A.6 Ceny

- Systém bude evidovat různé ceníky (kategorie cen)
- Ceník bude společný vždy pro celou podobjednávku
- Více podobjednávek může používat stejný ceník
- Systém může být provozován jako limitní či jako bez-limitní
- Ceník bude přiřazovat každé stravní položce vzorec (v případě limitního způsobu provozu bude vzorec konstantní, v případě bez-limitního bude záviset na každé vydávané položce).
- Položky bez definované ceny (bez vzorce) nebude možno strávníkům (patřícím do podobjednávek používajících tento ceník) vydat.
- Ceník bude možno měnit v čase, bude si uchovávat historii změn.
- Systém bude podporovat hromadné nastavování cen (pro celou skupinu stravy, pro všechny vybrané položky, apod.)
- Cena jídla je platná v době objednávky, v bezobjednávkovém provozu v době výdeje.
- Změna ceny položky po objednání stravy nebude mít vliv na objednanou stravu
- Systém nebude (nemůže) řešit situaci, kdy je cena stravy známa až po provedení objednávky. Pro kombinaci bezlimitního a objednávkového provozu je proto silně doporučen modul Normování.
- V bezlimitním bezobjednávkovém provozu bude systém podporovat fixaci ceny - ve stanoveném období se prodejní cena nezmění, pokud se neodchýlí o danou část od první či předchozí ceny v období. A. Příspěvky na stravu
- Systém umožní evidovat příspěvky na stravu.
- Příspěvek na stravu bude vázán na objednávku a stravu, strávník může využít stejný příspěvek u více položek stravy, více objednávek může používat stejný příspěvek
- Systém bude umět omezit poskytování příspěvku pouze na určitou stravu a na určitý počet poskytnutých příspěvků za nějaké časové období (nejvýše N denně, maximálně M-krát počet pracovních dní příspěvků v měsíci, ...)
- Příspěvek může být zadán vzorcem - může se jednat o fixní částku nebo o částku vypočtenou z ceny stravy.
- Příspěvek může hrazen třetí stranou nebo poskytnut ve formě slevy
- Systém bude umět více příspěvků na jeden odběr současně B. Konto a platby
- Veškeré transakce se zúčtují vůči kontu strávníka
- Konto lze dobíjet hotovostně nebo platební kartou.

- Konto lze dobíjet i převodem či inkasem, pokud bude k dispozici soubor z banky ve zdokumentovaném formátu, který bude obsahovat údaje nutné k určení konta a částky.
- Pokladna umí zpracovat operace v hotovosti, platební kartou a srážkou z konta strávníka.
- Uživatel má možnost nastavit si preferovaný způsob dobíjení konta - v případě inkasa i podmínku, za které dojde k jeho provedení.
- Systém nebude umožňovat omezení horní a hranice konta - toto musí řešit obsluha.
- Systém nebude umožňovat omezení dolní hranici konta - při přečerpání zobrazí systém varování, situaci však musí řešit obsluha (donutit strávníka k platbě).

A.7 Výdej

- Systém musí umožnit strávníkovi s platnou kartou a kontem odběr.
- Systém musí zablokovat odběr bez platné katy a konta.
- Při identifikaci je strávník přiřazen k podobjednávce, na základě toho je potom určena cena, příspěvek (příspěvky) na stravu a omezení příspěvků i výdeje.
- Rozhodnutí o provedení transakce navzdory omezení je možno nechat na obsluze - každé omezení bude mít nastavení "tvrdé" (omezení platí) nebo "měkké" (obsluha může odběr povolit).
- Omezení konta budou definována pro podobjednávku, může tedy záležet na tom, jakou kartu strávník při odběru použije
- Je-li na provoze definován jídelníček, systém umožní odebrat stravu z platného jídelníčku. V případě pokusu o odběr ze stravy která není v jídelníčku na takovém provoze, systém nechá rozhodnutí o povolení odběru na obsluze.
- Systém umožní evidovat odběry dodatečně (např. u zaměstnanců zařízení). Oprávnění zadávat odběr dodatečně bude mít pouze vybraná skupina uživatelů.
- Systém umožní oprávněnému uživateli stornovat vybraný odběr. Oprávnění stornovat odběr bude rovněž povoleno pouze vybrané skupině uživatelů.

A.7.1 Objednávkový provoz

- Systém bude umět pracovat v čistě objednávkovém režimu.
- Systém bude umět přijímat objednávky na stravu
- Systém přijme objednávku právě jen na stravu, která je v jídelníčku na zvolený den
- Systém umožní omezit počet objednávek (například počtem položek na skladě)
- Provedení objednávky strávníkem je považováno za okamžik uskutečnění zdanitelného plnění.

- Při provedení objednávky se cena stravy strhne z uživatelova konta.
- Systém zajistí kontrolu správnosti odběru a umožní vydat pouze stravu, kterou si strávník objednal.
- Systém může podporovat systém pro tisk stravenek s čárovým kódem, stejně jako zařízení pro jejich čtení
- Systém umožní zrušení objednávky
- Systém umožní nastavit dobu, po kterou lze objednávku zrušit.

A.7.1.1 Burza stravenek

- Systém umožní vést "burzu" stravenek - seznam objednávek, které nelze zrušit, ale které původní strávník nebude moci odebrat.
- Stravenku nabídnutou do burzy může jiný strávník koupit za cenu platnou pro něj (nového strávníka) v okamžiku nákupu.
- Neprodanou stravenku lze z "burzy" stáhnout.
- Systém nebude evidovat prodej stravenek přes "burzu".

A.7.2 Bezobjednávkový provoz

- Systém bude umět pracovat v čistě bezobjednávkovém režimu.
- V tomto režimu systém umožní výdej stravy strávníkovi, právě když to umožňují možná omezení (jídelníček, příspěvek)
- Systém umožní v případě nemožnosti vydat stravu z důvodů omezení buď o Nevydat stravu, nebo o Vydat stravu bez příspěvku (strávník hradí plnou cenu) o Vydat stravu podle ceníku pro hotovostní strávníky
- Správce může určit, že je možná pouze některá z uvedených variant C. Kombinovaný objednávkový a bezobjednávkový provoz
- V kombinovaném provozu zajistí systém jak funčnost objednávkového systému tak bezobjednávkového
- Navíc zajistí rezervaci jídel strávníkům, kteří si jídla objednali
- Je-li zbývající počet jídel menší, než počet dosud nevydaných objednaných jídel, zobrazí systém při pokusu o odběr objednané stravy strávníkem bez objednávky varování. O vydání či nevydání stravy rozhodne obsluha. Tato funkčnost bude omezená v off-line režimu.

A.7.3 Restaurační provoz

- Speciálním případem kombinovaného objednávkového provozu je provoz restaurační
- Obsluha v tomto případě přijímá objednávky stravy podle jednotlivých stolů
- Objednávku je možno zaplatit po částech (každý host svoji část účtu)
- Systém lze pro zrychlení dovybavit kuchyňskou tiskárnou objednávek

A.7.4 Offline provoz

- V případě přerušení spojení mezi databázovým serverem a terminálem bude systém bez přerušení pokračovat v práci (přejde do off-line režimu). Systém bude off-line pracovat v omezeném režimu.
- Funkce systému v offline režimu je zaručena do konce dne. Další provoz v tomto režimu není garantován a je silně nedoporučen.
- Systém uživatele informuje o přechodu do off-line režimu, ale nebude vyžadovat jeho součinnost.
- V off-line režimu budou dostupné funkce terminálu Pokladna a Výdejní místo.
- Funkce terminálu Výdejní místo bude v offline kombinovaném provozu omezena - při více výdejních místech nemůže systém hlídat počty rezervovaných (vydaných) jídel.
- Dostupnost funkcí jiných typů terminálů není při přerušení spojení s databází zaručena.

A.8 Správa

- Uživatel je každý, kdo pracuje se systémem.
- Obsluha je uživatel, který není strávník.
- Systém eviduje uživatele, kteří s ním pracují buď ve vlastní režii, nebo k tomu využívá systému Windows nebo SQL serveru
- Obsluhu dělí systém do skupin dle oprávnění
- Skupiny je možno vytvářet a měnit
- Uživatel je členem právě jedné skupiny
- Skupině i uživateli lze přidělit či odebrat oprávnění
- Oprávnění může měnit správce
- Některá oprávnění lze rozšířit i mimo terminál, na kterém obsluha pracuje
- Systém umožní uživateli právě ty činnosti, které odpovídají uživatelovým oprávněním v době pokusu o ně.

- Změna práv probíhá při přihlášení a připojení k databázi, časová historie se nevede. Kapitola 11. Vstupy a výstupy A. Vazba na sklady a normování
- Systém bude při výdeji používat modul Sklady
- Při výdeji bude systém odepisovat zboží ze skladu
- Systém bude umožňovat export údajů o pohybech stravy v XML formátu.
- Je-li nainstalován, bude systém používat modul Normování. Systém bude fungovat i bez modulu Normování
- Normování bude automaticky vytvářet položky na skladě, které bude možno dát do jídelníčku

A.8.1 Výstupní sestavy

- Systém umožní tisk
  - dokladu pro každý odběr
  - faktur příspěvků na stravu
  - faktur odběrů (u odběrů neplacených srážkou z konta)
  - sestav sumarizujících počty odběrů za zvolené období podle těchto kategorií:
    - ceník
    - firma
    - podobjednávka
    - typ karty
  - sestav sumarizujících tržby za zvolené období podle uvedených kategorií
- Systém bude obsahovat nástroj pro tvorbu uživatelských sestav

A.8.2 Import

- Systém bude schopen jednorázově či pravidelně importovat
  - Seznam karet
  - Seznam firem
  - Přehled nároků na příspěvky

A.8.3 Export

- Systém bude připraven pro export údajů do SAP - podmínkou pro funkčnost je součinnost implementátora SAP (poskytnutí formátu vět).
- Systém bude umožňovat export souhrnných sestav do .xml a .xls formátu
- Systém může umožňovat export těchto sestav i do databázové tabulky

A.9 Podpora

- Součástí předání systému je instalace u zákazníka
- Součástí předání systému je zaškolení uživatelů
- Reklamace bude řešit smlouva o nákupu systému (smlouva o dílo)
- Rozšíření systému se budou řídit (budoucí) smlouvou o podpoře

A.10 Evidované údaje

A.10.1 Položka odběru

- Zkratka
- Název
- Kategorie
- Je odebíráno?
- Je vydáváno?
- Jde-li o složený odběr (menu) o Seznam položek a variant
- Je-li jídelníček vytvářen ve více jazycích o Název v každém z použitých jazyků

A.10.2 Odběr

- Datum
- Karta
- Terminál
- Položka odběru
- Počet
- PodObjednávka
- Je zadáno dodatečně?
- Zadal
- Zadáno
- Je vydáno v určené sazbě DPH?
- SazbaDPH
- Je to odběr objednané stravy?

- Objednávka stravy
- Je to stornující odběr?
- Stornovaný odběr
- Jde-li o složený odběr (menu)
  - Seznam skutečně odebraných položek

A.10.3 Objednávka stravy

- Karta
- Položka jídelníčku
- Objednáno
- Provoz
- Datum odběru
- Je strava nabídnuta v burze?
- Je strava odebrána?
- Kdy odebráno
- Je zrušena?
- Kdy zrušeno

A.10.4 Jídelníčky

- Položka odběru
- Platí od
- Platí do
- Provoz
- Počet porcí
- Vydáno porcí
- Pokud jde o společné jídlo (jídlo vydávané pouze k jinému jídlu)
  - Seznam položek odběru, ke které je možno tuto položku jídelníčku vydat

A.10.5 Objednávky

- Popis
- Datum od
- Datum do
- Seznam podobjednávek

A.10.6 Podobjednávky

- Popis
- Datum od
- Datum do
- Ceník
- Typ identifikace strávníka o
  - Kartou
  - Osoba (nejvýše jedna podobjednávka na osobu)
  - Firma (nejvýše jedna podobjednávka na firmu)
- Seznam příspěvků
- Seznam omezení

A.10.7 Ceník

- Zkratka
- Název
- Poznámka
- Seznam položek

A.10.8 Položky ceníku

- Položka odběru
- Vzorec pro výpočet
- Platnost od
- Zadáno
- Zadal
- Zrušeno

- Zrušil I. Příspěvky
- Zkratka
- Název
- Vzorec
- Kategorie
- Je účtován později (např. u odběrů zaměstnanců, kdy se v okamžiku odběru neví, zda bude odběr v limitu)?

A.10.9 Omezení

- Počet odběrů
- Časový interval
- Poznámka
- Kategorie

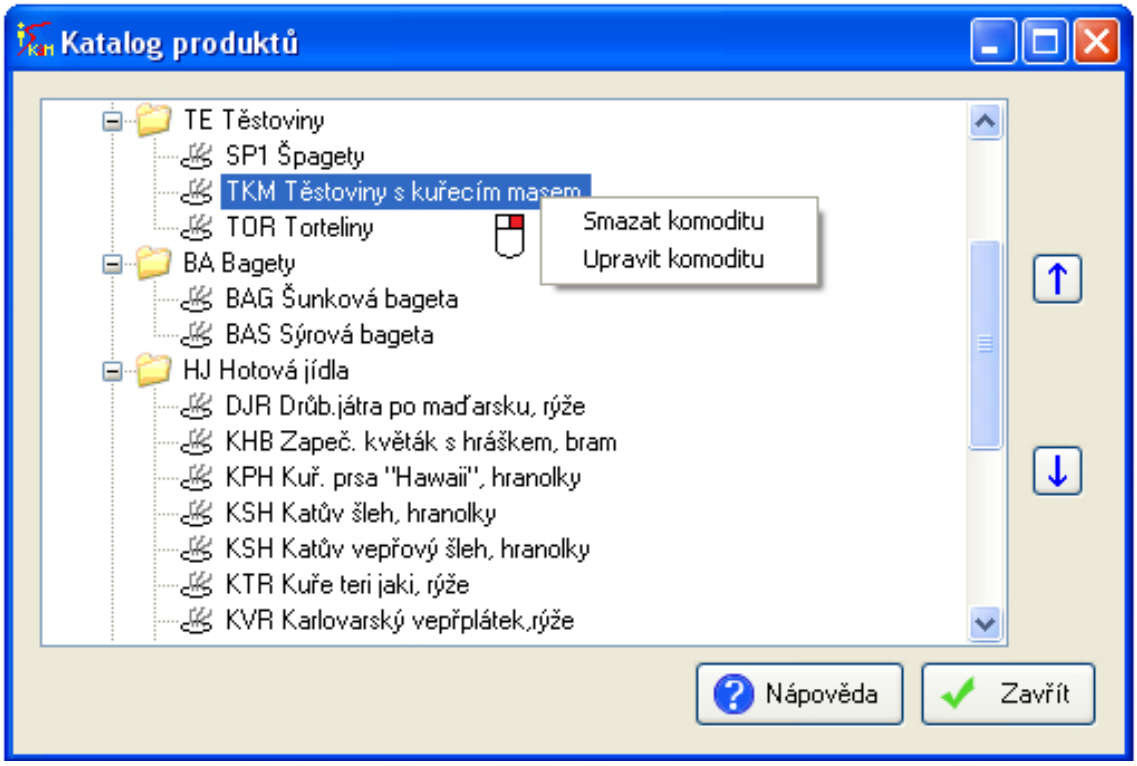
A.10.10 Vzorce

- Zkratka
- Název
- Vzorec
- Zaokrouhlení
- Poznámka

Dodatek B

Nápověda

B.1 Katalog produktů



Obrázek B.1: Katalog produktů

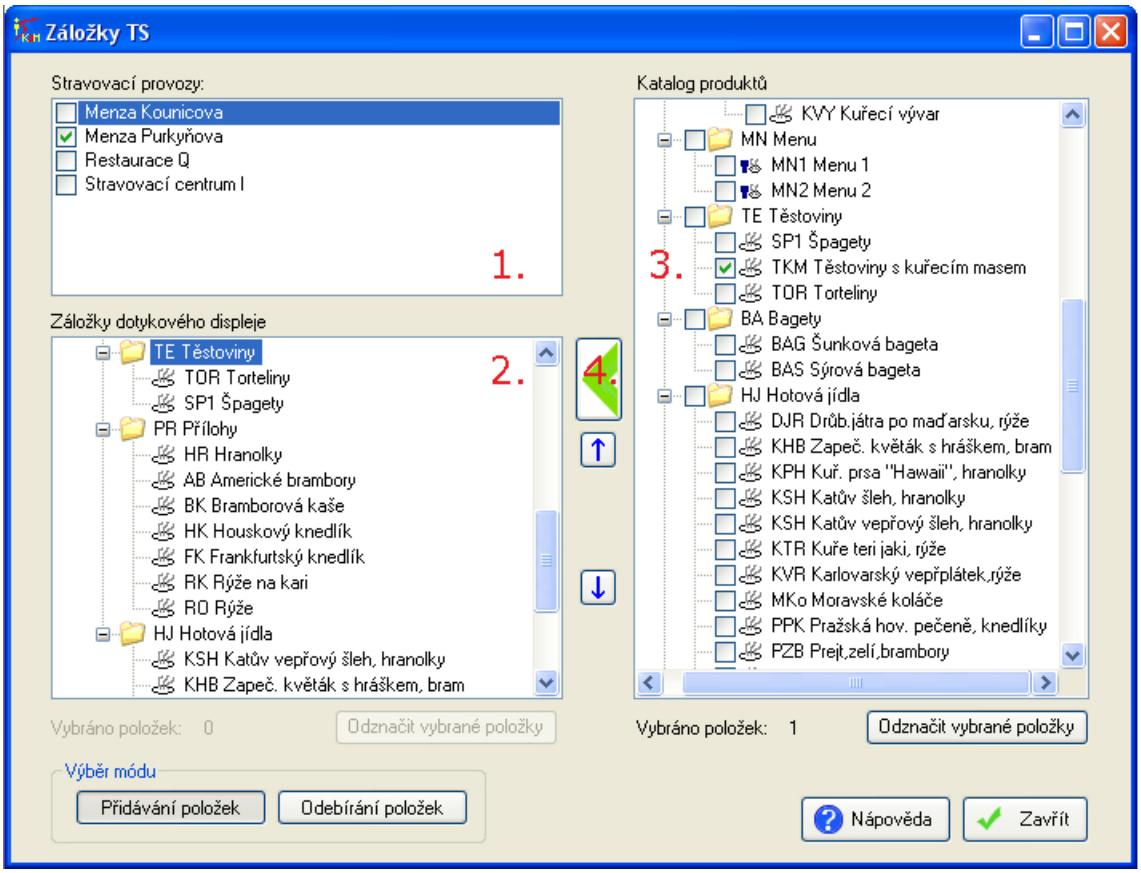
Tento dialog slouží pro správu veškerých komodit, které mohou stravovací provozy nabízet. Pro usnadnění přehlednosti jsou organizovány do stromové struktury. Každá složka může obsahovat buď:

- Další podsložky nebo
- jen komodity

Pro lepší přehlednost je kombinování složek a komodit uvnitř jedné složky zakázáno. Editace

- **Vytvářet** novou složku nebo komoditu stávající můžete kliknutím pravým tlačítkem myši na nějakou složku
- **Editovat** nebo **mazat** můžete obdobně pravým tlačítkem myši na vybranou složku nebo komoditu
- **Přesouvat** položky je možno uchopením levým tlačítkem myši a následným přetažením do jiné složky. Pro složky jsou v pravé části dialogu šipky, kterými je možno posouvat vybranou složku nahoru nebo dolů v rámci její nadřazené složky.

B.2 Záložky dotykového displeje



Obrázek B.2: Záložky dotykového displeje

Tento dialog slouží pro indivuduální přiřazení komodit k jednotlivým stravovacím provozům a také nastavení záložek pod kterými se budou komodity zobrazovat. Na různých provozech mohou být vybrány jiné skupiny komodit z katalogu produktů. Vybrat můžete také stejné komodity, které ovšem můžete přiřadit do jiných záložek.

Např.: Šunková pizza, může být v jedné menze pod záložkou hlavní jídla a v druhé pod záložkou pizzy

Použití

Záložky, které jsou v levé střední části(označeny číslem 2) můžete přidávat, upravovat, mazat a přesouvat podobným způsobem jako je tomu u dialogu Katalog produktů. Jediným výrazným rozdílem je to, že zde nelze vytvářet několikanásobné zanoření. Jednotlivé záložky tedy mohou být vloženy pouze do kořenové složky a nikam jinam.

Přidávání položek

Pro přidávání je potřeba mít zamáčknuto tlačítko "Přidávání položek", které se nachází vlevo dole. Chcete-li přidat nějaké položky, budete muset postupovat podle následujících 4 kroků(jsou vyznačeny i na obrázku):

1. Nejprve vyberete stravovací provoz. Můžete jich vybrat i více. V tom případě, nebude zobrazeno jejich složení, protože každý jej může mít jiné. Ovšem na samotné přidávání to nebude mít vliv.
2. V druhém kroku je potřeba vybrat cílovou záložku do které chcete přidávat komodity.
3. Ve třetím kroku vyberete komodity, které chcete přidat
4. Ve čtvrtém kroku stačí zmáčknout tlačítko se zelenou šipkou. Pokud jste předchozí tři kroky provedli správně, tak se komodity "nalijí" do levé části. Pokud již na vybraném provozu některé z vybraných komodit jsou, přiřadí se jen ty, které tam ještě nejsou.

Odebírání položek

Pro odebírání je potřeba mít zamáčknuto tlačítko "Odebírání položek", které se nachází vlevo dole. Princip je obdobný s tím rozdílem, že teď položky "vylíváme" zase zpět do prava. Takže zaškrtávátka, která Vám umožní vybrat více položek najednou, jsou nyní v levé části a tlačítko, kterým jste přidávali(označeno č.4) nyní funguje pro odebrání.

Jídelníček

Dialog jídelníček slouží pro vytváření jídelníčku. Jídelníček můžete upravovat pouze dnešní a budoucí. Do včerejších a starších již není možno zasahovat

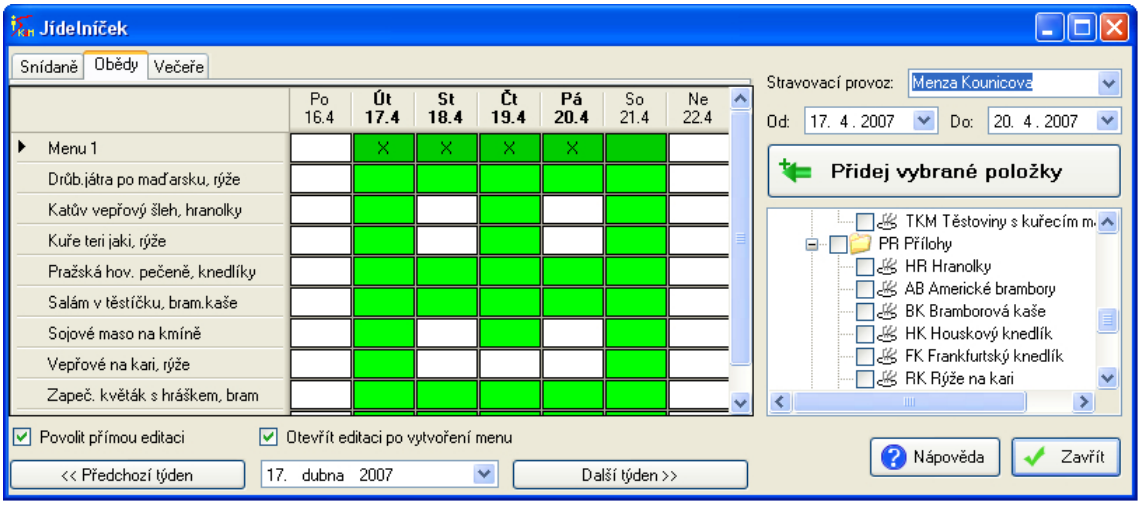
Použití

Doby výdeje

V levé horní části se nachází doby výdeje, které můžete pravým tlačítkem vytvářet a upravovat. Mazat je možné jen do té doby, dokud pro danou dobu není naplánována nějaká položka v jídelníčku. Nastavit si můžete také barvu, kterou se pak bude jídelníček zobrazovat.

Komodity

V pravé části můžete vidět strom, který odpovídá přiřazení komodit do záložek. Tento strom je shodný s tím, který jste nastavili v dialogu Záložky dotykového displeje. Proto zde budete mít pro plnění jídelníčku k dispozici jen tyto položky, které patří k vybranému provozu.



Obrázek B.3: Jídelníček

Jídelníček

Samotný jídelníček se nachází v levé střední části. Obarvené obdelníčky znamenají, že je položka na dané datum naplánována, bíle že nikoliv.

Chcete-li **přidat položky jídelníčku, které v něm ještě nejsou**, vyberte si tedy požadované komodity a nastavte rozsah datumů, do kterého chcete jídelníček naplnit. Datum můžete nastavit:

1. Roletkami nebo
2. pokud stisknete levým tlačítkem myši na počátečním datumu v jídelníčku a se stále stisknutým tlačítkem se dostanete na konečný datum, kde tlačítko myš uvolníte.

Chcete-li **editovat položky jídelníčku**, pak jsou zde opět dvě cesty:

1. Kliknete na řádek s položkou, kterou chcete upravit, čímž se Vám nastaví i její rozsahy, které můžete následně změnit a nechat uložit. Opět platí, že nelze manipulovat s jídelníčkem v minulosti, takže pokud tato položka jídelníčku začíná v minulosti, nelze měnit hodnotu "Od".
2. Další možností je zaškrnout si vlevo dole **Povolit přímou editaci** a pak můžete jednoduše editovat přímo zobrazený týdenní jídelníček. Kliknete-li na nějaký bílý obdelníček, je to totéž jako byste přidali položku určenou řádkem na datum určený sloupcem. Kliknete-li na obarvený obdelníček, funguje to obráceně, položka bude tedy smazána. Můžete dělat i výběry, tedy na jednom políčku zmáčknete tlačítko myši a na jiném jej uvolníte. Jestli se jedná o přidávání nebo mazání rozhoduje první políčko(tedy to, na kterém jste stiskli tlačítko myši)

Menu se zobrazují tmavší barvou než normální položky a jejich nastavování je odlišné od nastavování normálních položek. Je to kvůli tomu, že Menu1 v pondělí nemusí být totéž co Menu1 v úterý(kvůli odlišnému složení). není tedy možné klinout na řádek a změnit rozsah datumů. Chcete-li vybrané s nadefinovaným složením menu roztáhnout do budoucnosti, klikněte na něj pravým tlačítkem myši a vyberte datum.

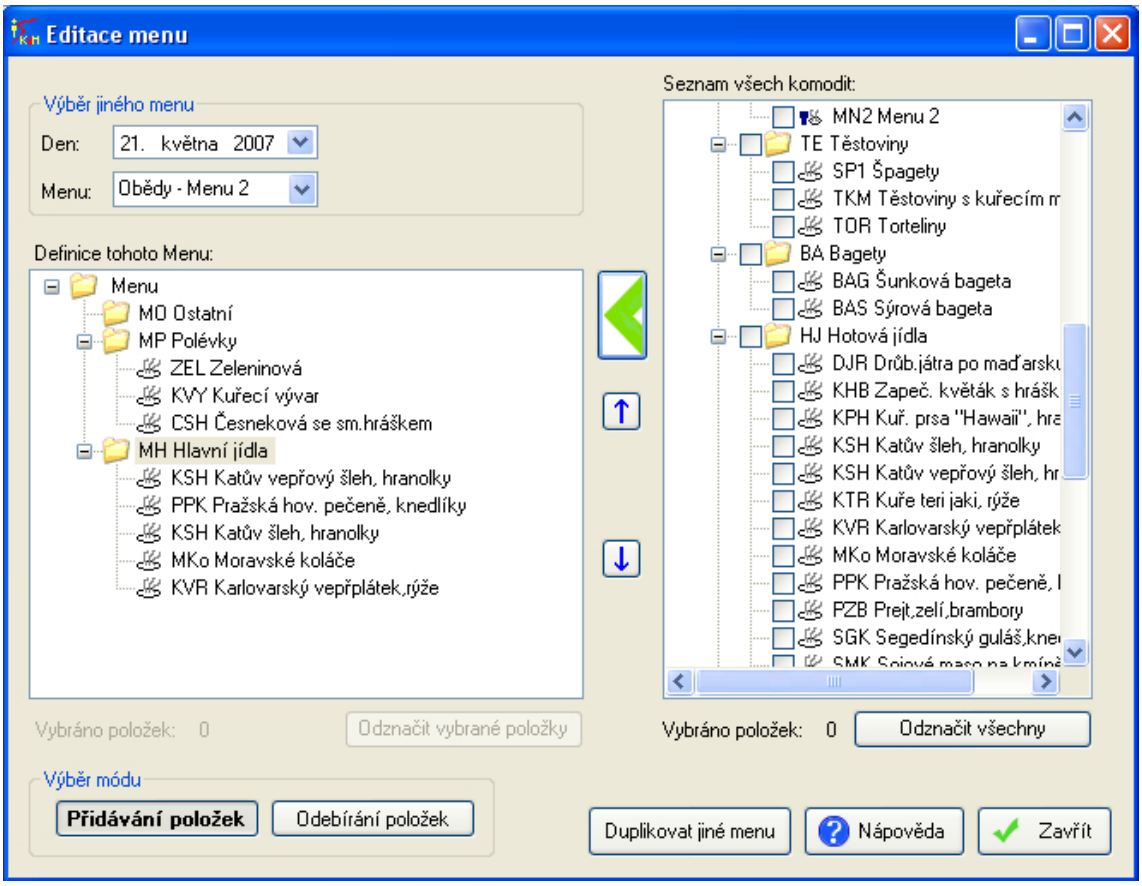
Definice menu

Pokud menu nemá nastaveno jeho složení, zobrazuje se v jídelníčku s písmenem X. Pokud má, můžete při najetí kurzoru na požadované menu vidět jeho složení. Definici menu můžete otevřít třemi způsoby:

1. Pravým tlačítkem myši na menu
2. Dvojklikem na menu. Ovšem pozor, pokud máte zaškrtnuto Povolit přímou editaci, pak bude dvojklik považován za 1 klik pro smazání a 1 klik pro vytvoření. Pokud je menu prázdné, tak to stále může splnit účel. Viz bod 3.
3. Zaškrtnutím políčka Otevřít menu po vytvoření, se menu otevře. Máte-li zaškrtnuty obě políčka, tedy i Povolit přímou editaci a 2x kliknete na menu, tak se stávající smaže, vytvoří se nové a to se otevře. Při současném zaškrtnutí obou těchto políček můžete vytvořením výběru prázdných polí editovat více menu najednou

Máte-li zaškrtnuto Otevřít menu po vytvoření, můžete vytvořením výběru prázdných polí editovat více menu najednou.

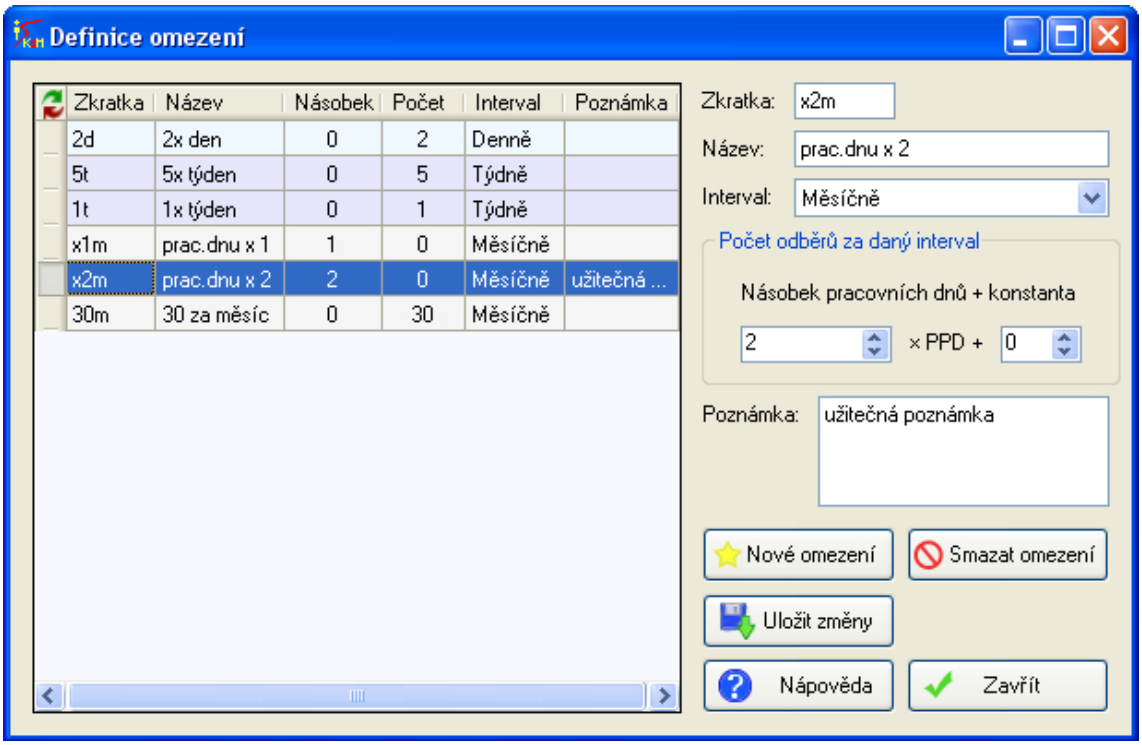
Definice menu



Obrázek B.4: Definice menu

Tento dialog je téměř identický s dialogem Záložky dotykového displeje. Navíc tento dialog obsahuje tlačítko pro duplikaci jiného již vytvořeného menu. Při duplikaci stačí vybrat které menu, chcete duplikovat a jeho složení se překopíruje do stávajícího. Původní položky budou samozřejmě odstraněny.

Definice omezení



Obrázek B.5: Definice omezení

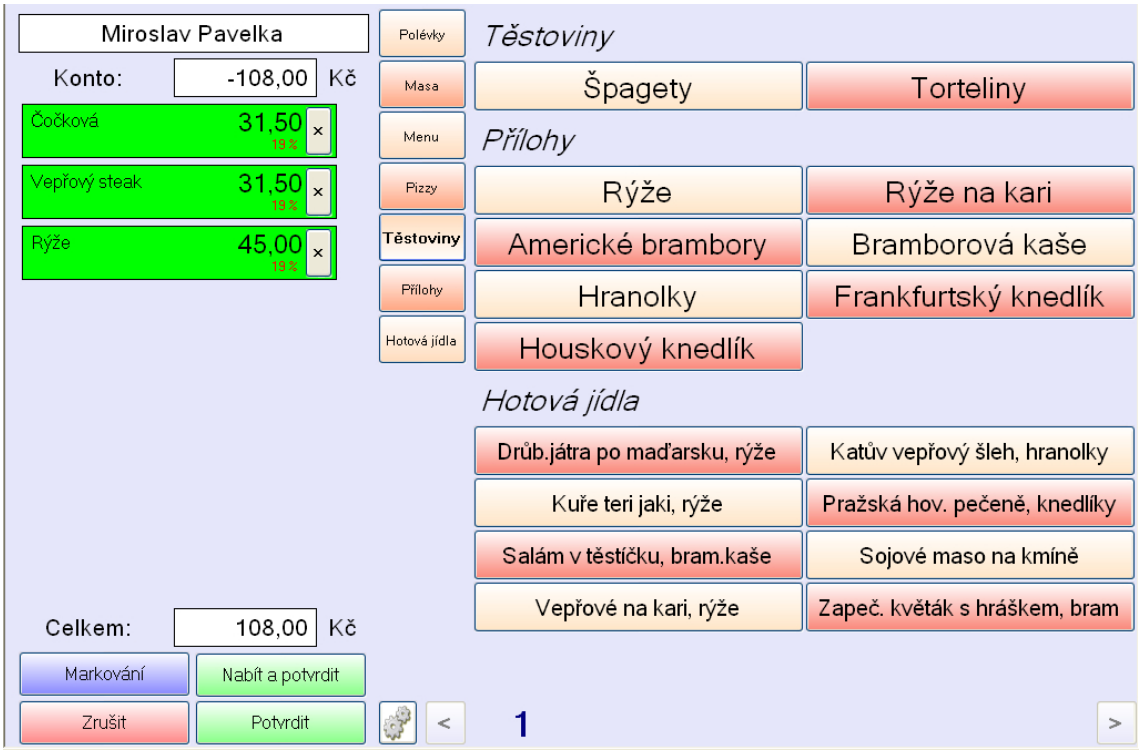
Tento dialog umožnuje definovat základní omezení, které později můžete přiřazovat k příspěvkům atp. Jsou 3 základní typy omezení:

- Denní
- Týdenní
- Měsíční

Ke všem třem typům můžete přiřadit kromě názvu, zkratky a poznámky taky číselnou konstantu, která bude určovat nárok v daném časovém období. U měsíčního intervalu ještě navíc můžete definovat násobek pracovních dnů. Pokud by tedy bylo v květnu 20 pracovních dnů a Vy si nastavíte v omezení do pole násobek hodnotu 1 a do pole konstanta hodnotu 5, pak v květnu bude nárok činit 25 jídel

Reálná omezení se poté budou skládat z 0-3 omezení(každé pro jiný časový úsek) a budou se navzájem ovlivňovat. Pokud bude mít strávník omezení 2/den a 5/týden a bude si brát každý den 2 jídla, tak ve středu už bude mít nárok jen na jedno(jídlo, příspěvek) a do konce týdne nic.

Výdej stravy



Obrázek B.6: Výdej stravy

Výdej stravy je dialog, umožňující na pokladnách s dotykovými displeji výdej stravy. Zatím se jedná o testovací podobu a chování. Velké množství úprav sice nabízí nastavení, ovšem v případě jakýchkoliv nedostatků doufám, že využijete zpětné vazby a pomůžete tak programu do stádia, kdy bude vyhovovat Vašim a představám.

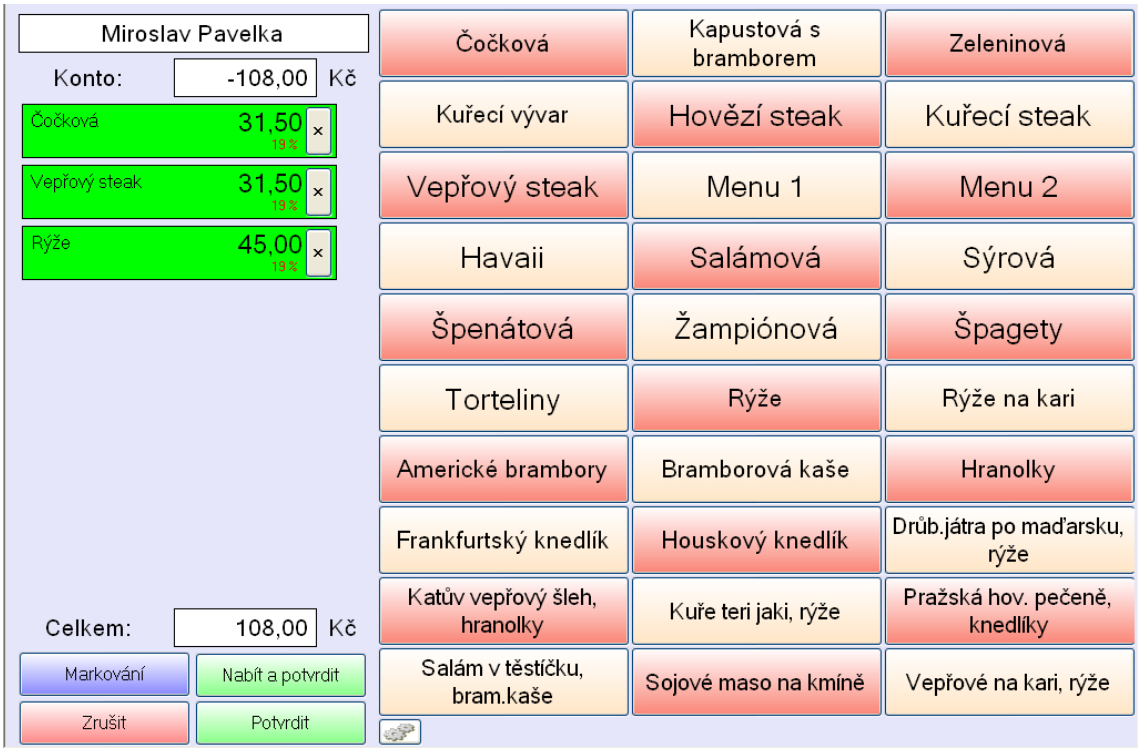
Nastavení

Roletka výdej umožňuje vybrat dobu výdeje. Pokud není hodnotu možno měnit, pak nejspíš není naplánováno více jídelníčku na tento den

Veškeré změny, které nastavením uděláte, má trvání pouze do dalšího spuštění a neovlivňuje nastavení na jiných pokladnách. Chcete-li si nastavení zálohovat, použijte tlačítko **Uložit své vlastní nastavení**. Tímto uložením se uloží barvy i zobrazení k vaší osobě. Opět to neovlivní jiné pokladny a navíc pokud se po Vás přihlásí někdo jiný, bude mít buď výchozí nastavení, nebo své uložené, nikoliv to vaše. Stejně tak, pokud Vy přijdete na jinou pokladnu, budete i tam mít své nastavení.

Při experimentování s barvami, pokud byste se chtěli vrátit výchozím barvám, můžete použít tlačítko **Obnovit výchozí hodnoty** nebo **Načíst své uložené nastavení**, pokud jste si nějaké uložili. Obsah jídelníčku se automaticky aktualizuje a ke znovunačtení dochází nejbližšší možné příležitosti, kdy není zahájen odběr. Atomatická aktualizace hledá rozdíly s načteným jídelníčkem. Pokud tedy není žádný načtený, nebude se automaticky aktualizovat, protože nebude vědět kterou dobu výdeje má kontrolovat.

Obrázek B.7: Výdej stravy



Obrázek B.8: Výdej stravy