VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

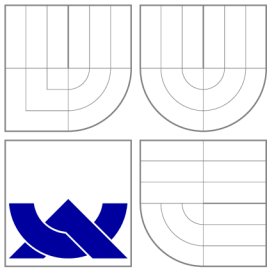
MOBILNÍ POKLADNÍ TERMINÁL

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

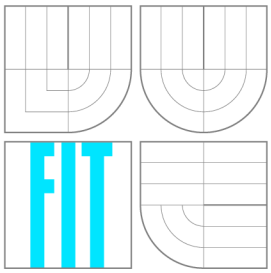
AUTOR PRÁCE
AUTHOR

PETR MALÝ

BRNO 2012



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

MOBILNÍ POKLADNÍ TERMINÁL

MOBILE CASH DESK

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

PETR MALÝ

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. RADOMÍR KUREČKA

BRNO 2012

Abstrakt

Bakalářská práce se zabývá návrhem a implementací systému mobilních objednávek v restauračním prostředí, který by obsluze umožnil vytvářet a zúčtovat objednávky přímo od stolu s možností natížit útratu na klientské konto po identifikaci klientskou kartou. Tento systém je implementován pro mobilní platformu Windows Phone verze 7.5.

Abstract

The bachelor thesis deals with the proposition and the implementation of system for mobile orders in restaurants, that would enable staff to create and to bill orders. Spending can be paid from customer's account after customer authentication. The system is implemented for mobile platform Windows Phone version 7.5.

Klíčová slova

Windows Phone 7, .NET, VB.NET, MVVM, mobilní zařízení, stravování

Keywords

Windows Phone 7, .NET, VB.NET, MVVM, Mobile Devices, Catering

Citace

Petr Malý: Mobilní pokladní terminál, bakalářská práce, Brno, FIT VUT v Brně, 2012

Mobilní pokladní terminál

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Radomíra Kurečky.

.....
Petr Malý
8. května 2012

Poděkování

Děkuji vedoucímu této práce panu Ing. Radomíru Kurečkovi za vstřícný přistup a konstruktivní náměty během tvorby této práce. Dále děkuji panu Ing. Petru Dubovi za odborné konzultace v oblasti vývoje software pro stravovací zařízení.

Obsah

Kapitola 1

Úvod

Rozmach mobilních zařízení s dotykovým ovládáním odhaluje nové možnosti ve všech odvětvích, se kterými běžně přicházíme do styku. Nejinak je tomu u stravovacích a restauračních zařízení. V době vzniku této práce se na tuzemském trhu začínají zvolna objevovat společnosti zaměřující se na usnadnění a zrychlení provozu takovýchto podniků. V této oblasti jsou z provozního hlediska jedny z nejdůležitějších faktorů rychlost a efektivita obsloužení zákazníka (měřítkem efektivity obsloužení zákazníka jsou finanční nároky na obsluhující personál a maximální počet obsloužených zákazníku za určitou časovou jednotku). Nasazení nových technologií tak může výrazně přispět ke konkurenceschopnosti podniku. Tato bakalářská práce se u stravovacích zařízení zaměřuje konkrétně na možnosti a způsoby realizace objednávek a plateb prostřednictvím mobilních terminálů.

Vybavení číšníků přenosnými terminály dnes už není vnímáno jako velký technologický převrat. I když mobilní terminály nepatří do standartního vybavení restaurací, je možné se s tímto způsobem zpracování objednávek setkat v řadě vyspělých zahraničních zemích, stejně tak i v České republice. Obdobná situace panuje i u plateb pomocí bezkontaktních karet, kdy je požadovaná částka odečtena z plátcova konta vedeného cílovou institucí. Tuto službu poskytují svým zákazníkům např. České dráhy a. s., které umožňují úhradu jízdenek a jiného zboží z elektronického konta klienta spárovaného se zákaznickou kartou (tzv. In-karta). Klient učiní požadovanou objednávku prostřednictvím terminálu vybaveného bezkontaktní čtečkou karet a po autentizaci In-kartou jsou z příslušného konta odečteny finanční prostředky k úhradě objednávky[1].

V této práci se tedy snažím o kombinaci a návrh nasazení obou zmíněných technologií do stravovacích zařízení. Podle průzkumu [4], který provedla a jehož výsledky koncem roku 2011 zveřejnila společnost MasterCard, mají více než tři čtvrtiny Čechů zájem o bezkontaktní platby. Hlavní výhody spatřují dotázaní v rychlosti, pohodlnosti a jednoduchosti plateb. Z průzkumu dále vyplynulo, že největší zájem o bezkontatkní platby je v supermarketech a obchodech s potravinami, veřejné dopravě a právě restauracích, kavárnách a fastfoodech.

Startovacím bodem této práce je stav, kdy máme stravovací zařízení s fungujícím informačním systémem. V podniku existuje centrální pokladna, přes kterou personál vkládá do systému veškeré objednávky zákazníků. Objednávka může být provedena dvěma způsoby. Prvním je tzv. rychlá objednávka, při které zákazník platí útratu neprodleně po odběru. Druhou možností je objednávka s otevřeným účtem, která je zpravidla vázaná na konkrétní stůl v podniku. Na tento účet mohou být během pobytu klienta v zařízení přidávány další

[1] Více na http://www.cd.cz/cs/vyhody-pro-cestujici/in-karta/elektronicka-penezenka-na-in-karte/

položky, které jsou vyúčtovány až na přání klienta. Cílem této práce je umožnit obsluhujícímu personálu vytvořit objednávku přímo od stolu tak, aby mohlo být ihned zahájeno její vyřizování. Přínos tohoto řešení je přímo úměrný rozlehlosti obsluhované plochy a množství obsloužených klientů.

Systém bude schopen akceptovat platby hotovostí nebo stravenkami. Navržené řešení navíc počítá s možností identifikace klienta pomocí RFID karty. Pokud má klient u daného podniku vedené konto, umožní mu systém uhradit odběr z prostředků uložených na kontě. Objednávky i platby budou realizovány pomocí mobilních terminálů. Mobilním terminálem se rozumí zpravidla kapesní počítač, mobilní telefon nebo jiné mobilní zařízení podobných rozměrů se srovnatelnými výpočetními schopnostmi.

První část práce obsahuje detailní analýzu požadavků na výstupní aplikaci, jsou zde také uvedeny příklady použití systému. Kapitola 3 se věnuje technologii radiofrekvenční identifikace a především návrhu aplikace a jejímu začlenění do již existujícího systému. Dále kapitola objasňuje volbu čtecího zařízení a mobilní platformy. V kapitolách 4 a 5 je popsána etapa implementace aplikace a jednotlivé fáze testování. V závěru práce jsou zrekapitulovány požadavky na aplikaci a jsou shrnuty dosažené výsledky.

Kapitola 2

Specifikace a analýza požadavků

2.1 Neformální specifikace

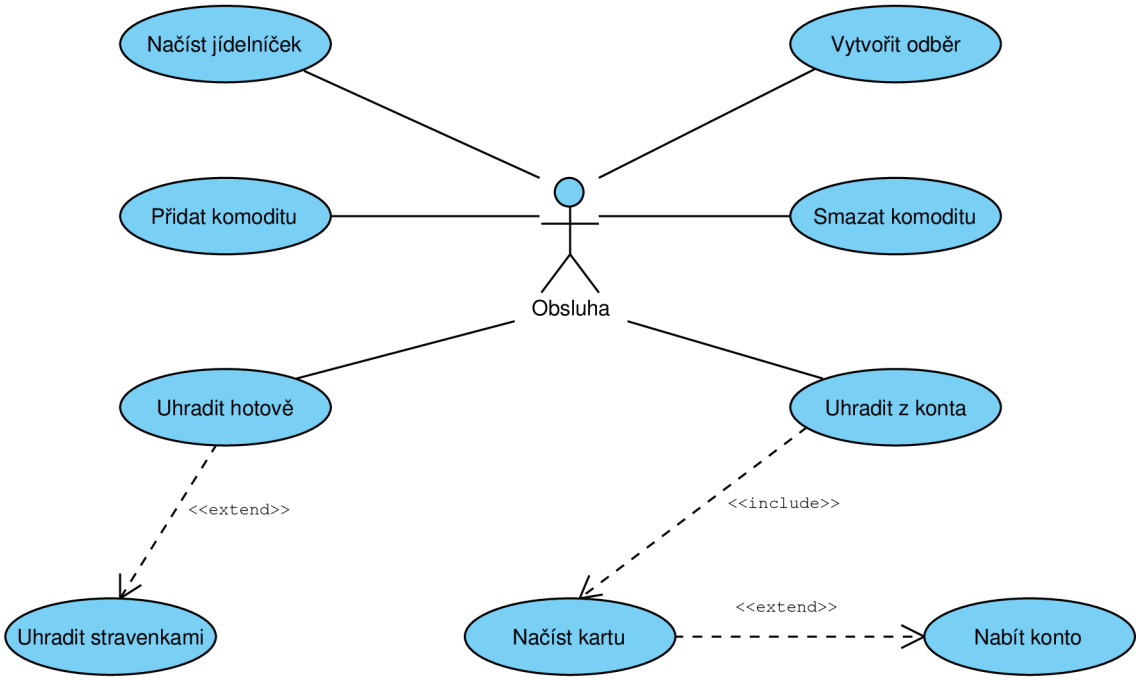
Na výstupu této práce se očekává kompletní návrh efektivního řešení objednávek a plateb pomocí mobilních terminálů v restauračním prostředí. Systém by měl umožnit zákazníkům platby z kont vedených u daného stravovacího zařízení prostřednictvím klientských karet. Součástí řešení bude obslužná aplikace implementovaná pro zvolenou mobilní platformu, která by měla umožňit využití *.NET Compact Framework*. Aplikace bude umožňovat personálu namarkování a zúčtování odebraných položek následujícím způsobem:

1. Při spuštění aplikace se ověří přístupové údaje uživatele, proběhne kontrola práv k výdeji stravy.
2. Načte se aktuální jídelníček s možností jeho obsah kdykoliv obnovit.
3. Zaměstnanec vytvoří objednávku, má možnost přidání a odebrání položek.
4. Po potvrzení objednávky zaměstnanec zvolí jeden ze dvou způsobů platby:
   a) platba kartou,
   b) platba hotově.
5. Zobrazí se informace o celkové výši odběru.
6. V případě dostatečného stavu konta se rovnou provede odečet a zobrazí se stav konta po odečtení odběru.
7. Pokud klient nemá na kontě dostatek prostředků, zobrazí se výše požadovaného doplatku. Při platbě v hotovosti bude výše doplatku rovna celkové výši odběru.
8. Záznam o vykonaných operacích bude ihned ukládán do databáze, aby bylo možné na požádání okamžitě vytisknout doklad o odběru.

Dále je třeba vzít v úvahu, že položky v jídelnčku se mohou během dne měnit, stejně jako se může měnit skladba nabízených produktů z aktuálního jídelníčku. Klientovi platícímu kartou by mělo být umožněno vložení prostředků na konto i nad rámec prováděného odběru. U hotovostního strávníka je třeba počítat s možností platby (nebo částečné platby) stravenkami.

2.2 Případy užití systému

Na základě informací z předešlé podkapitoly byl sestaven diagram případů užití (obrázek 2.1).



Obrázek 2.1: Diagram případů užití

Obsluha – Jediným aktérem, který bude s mobilním platebním systémem provádět operace, je obsluha stravovacího zařízení. K autentizaci a případně následné autorizaci dojde po přihlášení do systému. Tento případ užití není v diagramu uveden, stejně tak se v diagramu neobjevuje odhlášení ze systému. Samotný proces autentizace nemusí být nutně prováděn v rámci mobilní aplikace (identita uživatele může být ověřena na serveru, který aplikace pouze sdělí výsledek této operace).

2.3 Detaily případů užití

Níže se budu věnovat nejproblematičtějším případům užití, u nichž není zcela jednoznačný tok událostí a je třeba důkladnějšího pohledu na hlavní i alternativní toky.

Případ užití: Uhradit z konta
ID: UhraditZKonta
Účastníci: Obsluha Klient
Vstupní podmínky: 1. Obsluha stravovacího zařízení je přihlášena v aplikaci. 2. Je vytvořen odběr s nenulovým počtem komodit. 3. Klient má u stravovacího zařízení vedené konto. 4. Klient vlastní identifikační kartu.
Tok událostí: 1. Případ užití je zahájen volbou "uhradit z konta". 2. Obsluha načte klientovu kartu do systému. 3. Systém zobrazí aktuální stav klientova konta. 4. Ověření dostatečného kreditu na kontě. 5. Odečet kreditu z konta klienta, zaúčtování přijaté platby. 6. Systém zobrazí celkovou hodnotu odběru a stav klientova konta po uhrazení odběru. 7. Zobrazení zprávy o úspěšně provedené úhradě, uzavření odběru.
Následné podmínky: 1. Vytvořen záznam o přijaté platbě a odečtení kreditu ve stejné výši z klientova konta nebo zobrazení a vytvoření záznamu o chybě, která při operaci nastala.
Alternativní toky: 1. Přerušení procesu úhrady odběru obsluhou. 2. Nelze operovat s kartou. 3. Nedostatečný kredit na kontě. 4. Chyba v komunikaci se systémem.

Tabulka 2.1: Detail případu užití – uhradit z konta

Přerušení procesu úhrady odběru obsluhou – proces úhrady je ukončen. Obsluha může úhradu opakovat i jiným způsobem, nebo v odběru pokračovat.

Nelze operovat s kartou – zde může být příčin hned několik. Případ užití však může pokračovat. Následné událostí jsou přímo závislé na toku událostí v případu užití *Načíst kartu*.

Nedostatečný kredit na kontě – obsluze se o nastalé situaci zobrazí zpráva i s částkou, která do úplné úhrady odběru chybí. Obsluha má možnost pokračovat v případu užití přijetím chybějící částky hotově, nebo tento případ ukončit stornováním úhrady.

Chyba v komunikaci se systémem – v tomto případě bude záležet na rozsahu výpadku komunikace a architektuře systému, která bude předmětem dalších kapitol. Hlášení o chybě se zobrazí obsluze a chyba bude, pokud to bude možné, logována.

Případ užití: Načíst kartu
ID: NacistKartu
Účastníci: Obsluha Klient
Vstupní podmínky: 1. Obsluha stravovacího zařízení je přihlášena v aplikaci. 2. Klient vlastní identifikační kartu.
Tok událostí: 1. Případ užití se spustí volbou "načíst kartu". 2. Systém obdrží od čtečky identifikátor karty. 3. Ověření existence karty. 4. Ověření platnosti karty. 5. Identifikace klienta, který je vlastníkem karty. Zobrazení identifikačních údajů (jméno a příjmení). 6. Ověření existence klientova konta. 7. Zobrazení zůstatku na kontě klienta.
Následné podmínky: 1. Klient je jednoznačně identifikován, systém zobrazí jeho jméno a příjmení. 2. Obsluha má informaci o aktuálním zůstatku na klientově kontu.
Alternativní toky: 1. Čtečka karet nebyla schopna kartu načíst. 2. Karta je blokována. 3. Vlastník karty nemá vedené konto. 4. Chyba v komunikaci se systémem.

Tabulka 2.2: Detail případu užití – načíst kartu

Čtečka karet nebyla schopna kartu načíst – v tomto případě může být na vině mechanická závada. karty, nebo závada na čtečce. Obsluha má možnost zadat identifikační číslo karty ručně, případ užití pokračuje.

Karta je blokována – obsluze se zobrazí informace o blokaci karty. Každý pokus o operaci s touto kartou skončí neúspěšně.

Vlastník karty nemá vedené konto – systém zobrazí informaci o neexistenci konta. Takovýto klient nemůže provádět bezhotovostní úhrady. Pokus o úhradu skončí chybou.

Chyba v komunikaci se systémem – v tomto případě bude záležet na rozsahu výpadku komunikace a architektuře systému, která bude předmětem dalších kapitol. Hlášení o chybě se zobrazí obsluze a chyba bude, pokud to bude možné, logována.

Kapitola 3

Návrh aplikace

Před samotným návrhem systému, je třeba vzít v potaz umístění a roli alespoň základních síťových prvků, se kterými bude mobilní systém komunikovat. Dále je třeba zaměřit se na strukturu uložených dat, což vyžaduje vytvoření alespoň konceptuálního schématu relační databáze. To vše s ohledem na vybavení a možnosti zařízení, která jsou na trhu dostupná a mohou být pro účel mobilních plateb využita. Přehled existujících řešení je rovněž součástí této kapitoly.

Při návrhu systému se zaměřím na to, aby byly nasazení a integrace systému možné s minimálními zásahy do stávající síťové architektury, stejně tak bude kladen důraz na univerzálnost použití vzhledem k již existujícímu a funkčnímu informačnímu sytému, který stravovací zařízení používá. Dalšími aspekty jsou spolehlivost chodu a maximální možná odolnost proti výpadkům síťové komunikace.

3.1 Existující řešení

3.1.1 meet • U

Nasazení terminálového systému *meet • U* je možné v restauračních zařízeních stejně jako v hotelech, obchodech, muzeích, galeriích a dalších. Hlavním cílem tohoto řešení je umožnit koncovému uživateli atraktivní, interaktivní a snadný přístup k informacím. Verze *Tourist* a *Guide* zahrnují mobilní terminály s úhlopříčkou mezi 12 a 25 cm a systém, umožňující jejich použití jako interaktivních informačních průvodců. Pro obchody je nabízena verze *Shop*. Verze pro nasazení v restauracích, vinárnách, barech a klubech se nazývá *Restaurant*.

Řešení pro restauraci je postaveno na architektuře klient-server. Existuje jedna stanice, která plní roli serveru (v restauracích je touto stanicí zpravidla centrální pokladna). Stanice řídí a kooperuje se statickými terminály na stolech a také s mobilními terminály obsluhy. Tato komunikace probíhá prostřednictvím lokální Wi-Fi sítě. Je tedy třeba zajistit dostatečné pokrytí prostor podniku. Aktualizace software proběhnou nahráním nové verze na server, ten ji pak distribuuje na všechny terminály.

Hardwarové a softwarové řešení *meet • U* pro restaurace je zaměřeno především na statické terminály u stolů, prostřednictvím kterých vytváří objednávky sami zákazníci. Poskytují jim i další funkce jako komunikaci s obsluhou nebo barem, s ostatními stoly v podniku, připojení k internetu a další. Primární funkcí terminálů však nadále zůstává nabídnutí zákazníkovi aktuálního jídelního a nápojového lístku. Terminály u stolů navíc umožňují provozovateli podniku informovat zákazníky o právě probíhajících (např. akce typu Happy hours) nebo chystaných akcích.

Obrázek 3.1: Digitální jídelníček v systému meet•U na mobilním terminálu [8]

Nasazení mobilních terminálů je v tomto případě doporučeno pro rozsáhlé provozy nebo letní zahrádky. Číšník je schopen přímo od stolu odesílat objednávky jídel do kuchyně a objednávky pití na bar. Mobilní terminály také umožňují zadávat informace o platbách zákazníků. Doklad o platbě může být vytisknut na tiskárně na baru po zadání příslušného požadavku z kteréhokoliv mobilního terminálu. Kromě zrychlení komunikace mezi obsluhujícím personálem s barem a kuchyní slibuje toto řešení také úsporu pracovních sil. Informace o systému *meet•U* byly čerpány z oficiálních webových stránek [8].

3.1.2 SmartPOS

Jedná se o univerzální pokladní systém vyvíjený pro prostředí Microsoft Windows. Software je nabízen ve dvou verzích. Verze *Prodejna* je určena pro maloobchodní prodej zboží a služeb. Verze zaměřená na stravovací zařízení se nazývá *Gastro*.

Hlavním cílem pokladního systému *SmartPOS Gastro* je urychlení obsluhy zákazníka, čímž by mohl být navýšen počet vyřízených objednávek. Systém *SmartPOS Gastro* pracuje v restauračních zařízeních především v součinnosti s pevnými terminály s dotykovou obrazovkou. Toto řešení umožňuje personálu komunikaci se systémem z jednoho, nebo několika pevně daných centrálních míst. Možnost mobilních plateb a objednávek nabízí až rozšíření *SmartPOS Pocket*. Tato rozšířená verze je určena k nasazení na mobilní bezdrátové terminály na bázi PDA. Systém je schopen evidovat platby v hotovosti (i v cizí měně), stravenkami, nebo natížit útratu na hotelové či předplacené konto klienta. Z oficiálních stránek systému [7], z nichž byly čerpány informace v této podkapitole, není zcela zřejmé, jakým způsobem dochází k autentizaci klienta. Systém dále umožňuje zadávat procentní či absolutní přirážky a slevy k cenám objednávek, zasílat do kuchyně poznámky

k objednávkám (např. pokud má zákazník na objednávku nějaký nestandartní požadavek) a na základě oprávnění stornovat celou objednávku, nebo pouze jednotlivé položky v ní obsažené. K systému *SmartPOS* je možné využít volitelný modul *Sklad*, který umožňuje vést evidenci prodávaného sortimentu.

Informace o komunikaci mezi jednotlivými terminály, způsobu výměně dat a způsobu synchronizace v off-line režimu jsou předmětem obchodního tajemství a nejsou tak k tomuto systému veřejně dostupné.

3.1.3 Mobuyle

Systém *Mobuyle* vyvinutý společností Heartland Payment Systems představuje řešení mobilních plateb bankovními kartami. Jedná se čistě o platební systém. Jeho nasazení do stravovacích provozů by vyžadovalo existenci vlastního interního systému, který by pokrýval mimo jiné i oblast mobilních objednávek. Přínosem tohoto systému je umožnění klientovi bezhotovostní platbu prostřednictvím bankovní karty přímo od stolu. Příslušná částka bude strhnuta z bankovního konta klienta. Není tedy zapotřebí, aby měl klient u daného zařízení vedeno další konto.

Předpokladem pro toto řešení je existence mobilního zařízení s operačním systémem Android, pro který je aplikace *Mobuyle* vyvinuta. Mobilní zařízení musí být také vybaveno audio konektorem Jack (velikost konektoru není specifikována), do které se připojí čtečka karet přímo od společnosti Heartland. Přenosný pokladní terminál pak vypadá podobně jako zařízení na obrázku 3.2. Pro šifrování přenášených dat využívá aplikace šifrovací algoritmus AES. Pokud se aplikaci nepodaří zdárně dokončit transakci z důvodu nedostupnosti spojení se vzdáleným serverem, je transakce uložena a odeslána ve chvíli, kdy je spojení obnoveno. Informace k tomuto pokladnímu systému byly čerpány z webových stránek společnosti Heartland [1].

Obrázek 3.2: Mobilní pokladní terminál s aplikací Mobuyle [1]

3.1.4 Shrnutí

Každý z představených systémů nějakým způsobem zasahuje do oblasti mobilních objednávek nebo plateb. Společně odpovídají na všechny požadavky, kladené na výstupní aplikaci této práce, kterou si tedy lze ve finální podobě představit jako kombinaci zde uvedených řešení, tzn. aplikaci schopnou pracovat s objednávkami a poskytující zákazníkovi možnost bezhotovostní platby, která bude ale na rozdíl od systému *Mobuyle* provedena z konta klienta vedeného u daného stravovacího zařízení (podobně jako natížení objednávky na hotelový účet klienta u systému *SmartPOS*).

3.2 Radiofrekvenční identifikace

Technologie radiofrekvenční identifikace (dále jen RFID) je v současné době nasazena v mnoha odvětvích. S úspěchem je využívaná např. k automatizaci vybírání nejrůznějších poplatků, monitorování pohybu zvířat a v neposlední řadě k identifikaci osob, ke které je využita i v rámci této práce. Předpokladem je, že každý klient, který má u stravovacího provozu vedeno konto, vlastní bezkontaktní RFID kartu s unikátním číslem čipu. Na základě autentizace klienta pomocí karty, je danému klientovi umožněno čerpat finanční prostředky z příslušného konta. Při výběru čtecího zařízení je třeba přihlédnout k používanému typu karty a dbát na kompatibilitu mezi kartou a čtečkou. Jako příklad uvádím velmi často používané identifikační karty značky MIFARE od společnosti NXP Semiconductors, jejichž prodej celosvětově přesáhl 1 miliardu kusů. Produktovou řadu MIFARE tvoří několik typů karet, které se liší v různých parametrech. Odlišnosti nastávají u typu standardu, kterému karta vyhovuje, velikosti paměti nebo stupni zabezpečení uložených informací [6].

3.3 Tenký a tlustý klient

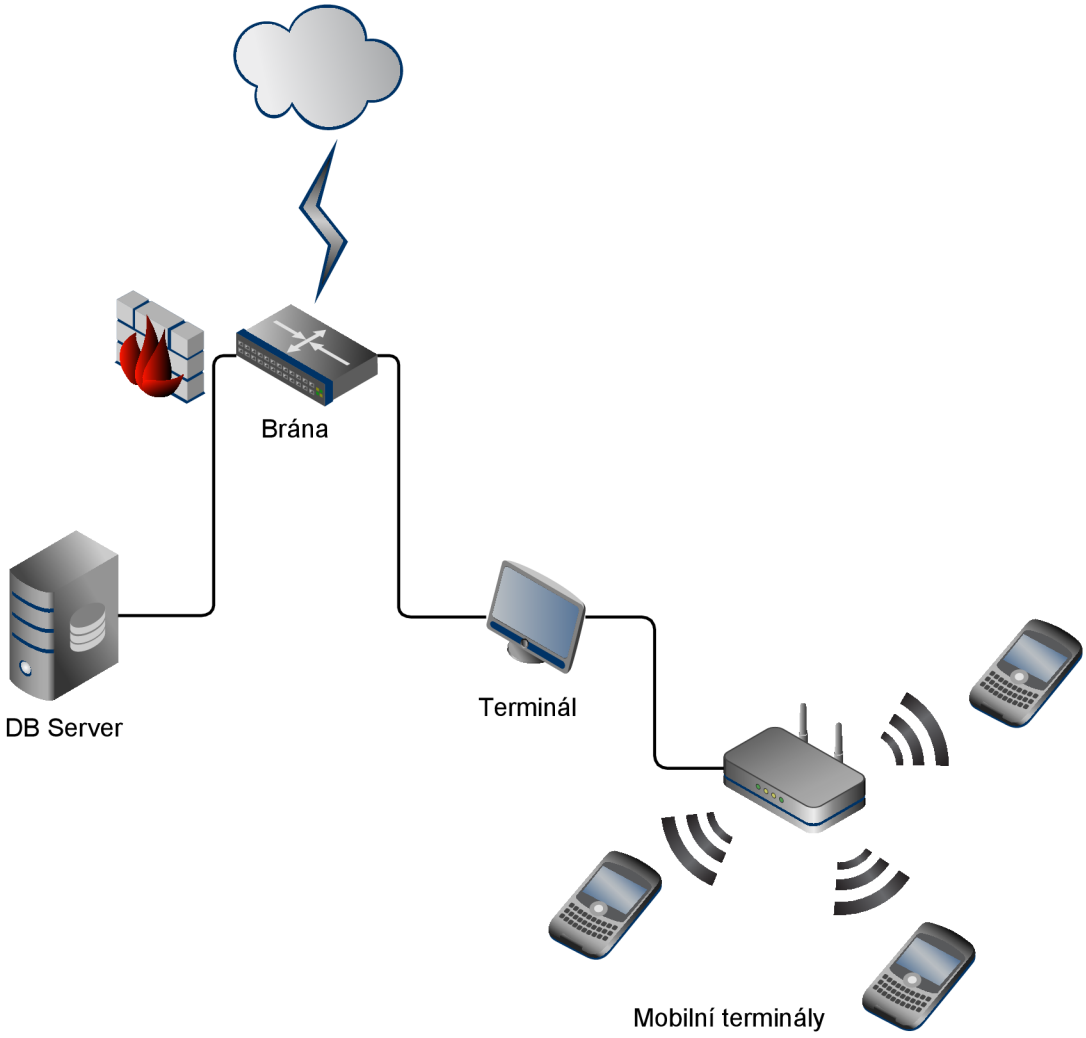
Podle rozložení výpočetního procesu, uložení aplikační logiky a dat můžeme klienta, kterým bývá obvykle počítač, označit jako tlustého nebo tenkého. Tenký klient zpravidla přistupuje k údajům přes webový prohlížeč. Může jím tak být zařízení od nejvýkonnějšího serveru po mobilní zařízení s podporou WAP[1]. Hlavní výhodou tenkého klienta je, díky využití webového prohlížeče, podpora širokého spektra zařízení a platforem. Jako nevýhodu lze označit fakt, že tenký klient z principu nemůže pracovat v off-line režimu. Dalšími nevýhodami tenkého klienta jsou malé využití výkonu lokálního zařízení, omezené zobrazovací schopnosti při tvorbě uživatelského rozhraní a omezená nebo v horším případě zcela nemožná komunikace s periferiemi.

Tlustý klient (v angličtině označován jako *rich client*) se nazývá klientská aplikace běžící na lokálním zařízení, kde je také soustředěn hlavní výpočetní proces a celá aplikační logika. Může tak naplno využít možností, které nabízí hardware zařízení. I tlustý klient však může využívat údajů a služeb poskytnutých serverem. Tlustý klient může běžet v režimu on-line, stejně jako v off-line režimu. On-line režim může být využit například k synchronizaci dat se serverem nebo jejich replikaci. Hlavní nevýhodou takového řešení je, že výsledná aplikace se musí přizpůsobit cílové platformě [2].

[1] *Wireless Application Protocol* – protokol umožňující bezdrátový přístup mobilních zařízení k datům a službám jiných zařízení.

3.3.1 Navrhované řešení

Na tomto místě zhodnotím na základě poznatků o aplikacích typu tenký a tlustý klient všechny přínosy a omezení, které plynou z jednotlivých řešení a jsou pro návrh systému mobilních plateb relevantní. Ze zvoleného řešení budou vycházet i omezení, která se projeví až v pozdějších fázích životního cyklu softwaru jako je provoz systému a jeho údržba včetně případných aktualizací. Navržené řešení proto přihlíží i k těmto aspektům.



Obrázek 3.3: Konceptuální síťová architektura

Obrázek 3.3 zjednodušeně znázorňuje možnou architekturu podnikové sítě stravovacího zařízení. Výchozím stavem je fungující systém pro výdej stravy, jehož kontaktním bodem je lokálně instalovaný statický terminál. Cílem návrhu je představit efektivní řešení rozložení výpočetního procesu mezi statickým a mobilním terminálem. Dále je třeba pro mobilní terminál zvolit vhodné umístění datového úložiště.

Tlustý klient

+ Možnost pracovat v off-line režimu
+ Rychlá odezva aplikace
+ Snadný a časově méně náročný vývoj
− Vyšší nároky na HW zařízení, což povede k vyšší pořizovací ceně zařízení
− Komplikovaná aktualizace, nutnost podpory komunikace různých verzí
− Obtížná přenositelnost

Tenký klient

+ Nízké nároky na HW klienta, větší škála využitelných zařízení
+ Snadné nasazení, správa a aktualizace
+ Přenositelnost
− Obtížná komunikace s místními systémovými zdroji
− Pomalejší odezva
− Větší zátěž serveru
− Omezené možnosti při tvorbě uživatelského rozhraní

Možnost pracovat v off-line režimu by vyžadovala autonomnost každého jednotlivého mobilního zařízení tzn. nezávislý provoz mobilního zařízení bez závislosti na ostatních mobilních a jiných zařízeních. Tímto řešením by se zcela vyřešily problémy spojené s výpadky síťové komunikace, v praxi je však toto řešení nerealizovatelné. Předpokládá se, že stravovací zařízení bude využívat několik mobilních terminálů najednou. Na každém zařízení by tak musela být uložena buď úplná databáze, nebo minimálně ta část, potřebná k výdeji stravy a vztahující se k danému výdejnímu místu (v případě několika výdejních míst pod centrální správou jako např. restaurační řetězce, vysokoškolské menzy, závodní jídelny). Toto řešení je neefektivní hned z několika důvodů. Možnost uložit databázi na každý jednotlivý terminál by výrazně zvýšila nároky na paměťovou kapacitu. Další překážkou by byla velká režie spojená s udržováním konzistentího stavu databází na všech zařízeních. Pokud by navíc zařízení měla po dobu provozu vzájemně kooperovat (tato situace může nastat např. pokud by měl být odběr vytvořený na jednom terminálu zúčtován terminálem jiným), znamenalo by to téměř nepřetržitou synchronizaci.

Lepším řešením se tedy jeví sdílené datové úložiště, které bude společné pro všechny mobilní terminály v rámci jednoho výdejního místa. Sdíleným úložištěm může být lokální server, stanice nebo centrální pokladna, nad kterými mohou být další datová úložiště vyšších úrovní, o která už se však mobilní zařízení nemusí zajímat. Mobilní zařízení bude tedy ve spojení pouze s lokální stanicí. Tím odstíníme celou zbývající podnikovou hierarchii. Další výhodou, kterou nám toto řešení poskytuje je odolnost proti výpadkům podnikové sítě. Pokud by lokální terminál nemohl být datovým zdrojem pro mobilní zařízení z důvodu, že sám využívá jiné datové úložiště v síti, musely by i mobilní zařízení přistupovat k tomuto úložišti a to buď přímo, nebo prostřednictvím lokálního terminálu. V obou případech by se tak staly zcela závislými na stabilitě podnikové sítě. Případný výpadek by zcela

znemožnil provoz všech terminálů a způsobil ztráty provozovateli. Komunikace mezi mobilními zařízeními a lokálním terminálem bude probíhat bezdrátově za použití jednoho nebo několika směrovačů instalovaných takovým způsobem, aby poskytovaly dostatečné pokrytí celého obsluhovaného prostoru. Tato komunikace bude naprosto nezávislá na síťové komunikaci lokálního terminálu s případnými zařízeními vyšší úrovně. Výpadek tohoto spojení by neměl vliv na funkci mobilních platebních terminálů.

Lze očekávat, že požadavky na funkcionalitu aplikace se budou v čase měnit. Na tyto nové požadavky bude třeba flexibilně reagovat formou aktualizací nebo doplňků. Z tohoto pohledu se jako ideální řešení jeví aplikace koncipovaná jako tenký klient, která by na mobilních zařízeních plnila pouze zobrazovací funkci. Toto řešení ale znemožňuje požadavek na čtení klientských karet pomocí vestavěné nebo externí RFID čtečky (viz podkapitola 3.4). Z tohoto důvodu bude aplikace implementována pro konkrétní cílovou platformu, což mimo jiné omezí její další přenositelnost.

Navrženým řešením tedy bude klientská aplikace vhodně kombinující výhody tlustého a tenkého klienta. Takové řešení bývá označováno jako chytrý, střední nebo také hybridní klient. Vyjdeme-li ze základní podoby třívrstvé softwarové architektury (datová, aplikační a prezentační vrstva), můžeme konstatovat, že aplikace na mobilním zařízení bude obsahovat prezentační vrstvu a část aplikační vrstvy, zatímco k datové vrstvě bude přistupovat prostřednictvím lokální sítě.

3.4 Mobilní terminál

Při výběru mobilního terminálu je nezbytné zohlednit každý jednotlivý požadavek, který na funkcionalitu a vybavení zařízení klademe. Pro synchronizaci a výměnu dat je žádoucí, aby byl terminál schopen komunikovat přes bezdrátové rozhraní. Dále musí být terminál snadno uchopitelný a ovladatelný. Číšníci jej nejspíše budou mít při sobě po celou otevírací dobu podniku, je tedy třeba zohlednit i rozměry a váhu zařízení. Dalším neméně důležitým faktorem jsou pořizovací náklady.

Protože má být terminál schopen komunikovat s bezkontaktními identifikačními kartami, je třeba věnovat zvýšenou pozornost výběru čtečky. Čtečka je zařízení umožňující obousměrnou komunikaci s kartou, jehož úkolem je poskytnout kartě zdroj napětí a synchronizačního signálu, signalizovat přítomnost karty a vytvořit komunikační rozhraní mezi kartou a řídící aplikací. Identifikačních karty ovšem mohou využívat rozdílná frekvenční pásma nebo různé komunikační protokoly. Čtečky tak zpravidla nejsou kompatibilní s libovolným typem karty a jejich výběr přímo závisí na typu používaných identifikačních karet. Pokud se budeme snažit o to, aby vybraný terminál byl na typu identifikačních karet co nejméně závislý, bude třeba výběr terminálu rozdělit do dvou částí:

1. čtečka RFID karet;
2. samotné mobilní zařízení.

Mobilní zařízení, která obsahují integrovanou RFID čtečku, jsou ve většině případů uzpůsobena do náročnějších prostředí, než je to restaurační (např. logistické sklady). S tím souvisí jejich zvýšená robustnost (jedna z možných podob zařízení je zobrazena na obrázku 3.4). Pro číšníky by takovéto mobilní terminály byly hůře ovladatelné, přenositelné a stěžovaly by jim další běžné pracovní úkony. Nicméně odolnost terminálu je bezpochyby dalším z kritérií výběru. Je na zvážení pořizovatele, zda investovat do většího počtu méně odolných

zařízení, jejichž služba končí prvním vyklouznutím z ruky číšníka, nebo nakoupit odolnější zařízení, u kterých se očekává znatelně delší životnost a také vyšší pořizovací cena.

Integrované čtečky s sebou navíc nesou pro provozovatele podniku omezení při výběru a případné výměně klientských karet. V situaci, kdy by se provozovatel rozhodl přejít z jednoho typu karet na typ jiný, ať už by důvodem byl požadavek na vyšší úroveň zabezpečení, vysoká poruchovost karet aj., který by nebyl současně využívanými zařízeními podporován, byla by tato zařízení nadále nepoužitelná.

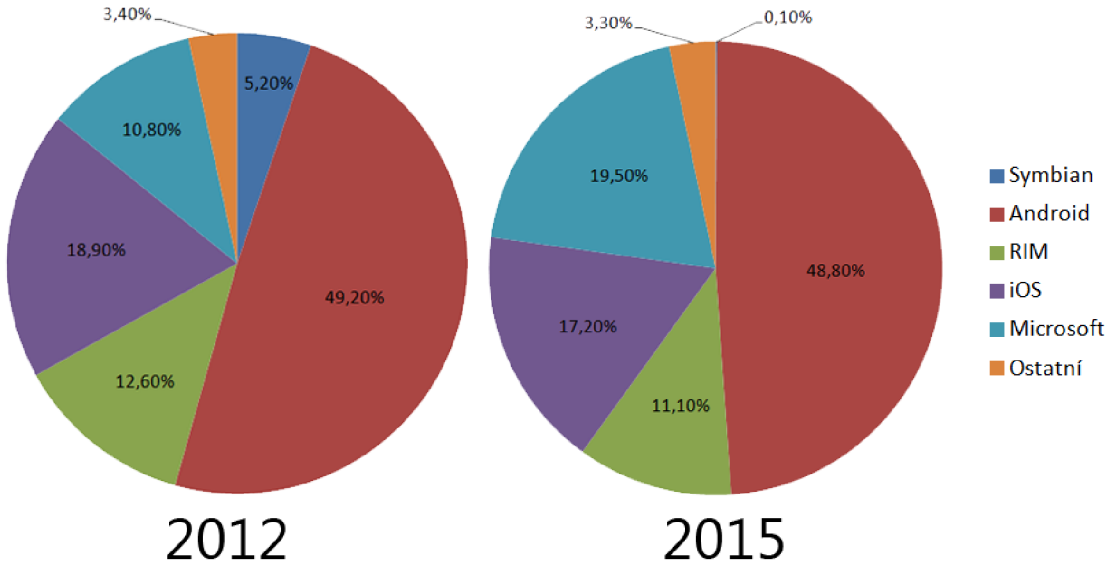
Tímto faktem byla inspirována snaha o rozdělení výběru terminálu do dvou částí. Externí RFID čtečka poskytuje při výběru samotného mobilního zařízení volný prostor, který je omezen pouze způsobem komunikace mezi zařízením a čtečkou. Čtečka může být s mobilním terminálem spojená buď přímo, nebo bezdrátově přes rozhraní Bluetooth (příklad Bluetooth čtečky je na obrázku 3.4). Jedním z možných způsobů přímého spojení je RFID čtečka připojitelná do SD slotu. Zde by ovšem s největší pravděpodobností vznikala potřeba použití redukce, protože současným trendem jsou u mobilních zařízení sloty miniSD a microSD a tím by se snižovala odolnost proti vnějším vlivům a manipulovatelnost se zařízením.

Bezdrátová čtečka svými rozměry umožňuje, aby ji měl číšník po celou pracovní dobu při sobě (např. v náprsní kapse). Pořizovací náklady tohoto řešení jsou oproti zařízením s integrovanými čtečkami pětinové (jedná se jen o rámcový odhad, skutečný rozdíl cen se bude odvíjet od ceny zvoleného mobilního zařízení, které bude komunikovat s externí čtečkou). Hlavní výhodou je ale vzájemná nezávislost obou součinných prvků. V případě přechodu na jiný typ karet, stačí vyměnit pouze čtečku a mobilní zařízení může být dál využíváno. Stejně tak, nastane-li potřeba výměny mobilního terminálu, může být nadále využívána stejná čtečka. Mobilní zařízení s integrovanými čtečkami navíc ani zdaleka nereagují na nástup nových mobilních operačních systémů tak pružně jako standardní mobilní přístroje, volba mobilní platformy by tak byla citelně omezena.

Obrázek 3.4: Terminál s integrovanou RFID čtečkou (vlevo) a Bluetooth RFID čtečka[2]

3.5 Výběr mobilní platformy

Pro implementaci jsem zvolil aplikační platformu Windows Phone (dále jen WP) od společnosti Microsoft, která poskytuje dostatečné nástroje pro realizeci aplikace, jež je výstupem této práce. Ač se podíl WP na trhu mobilních operačních systémů odhaduje na začátku roku 2012 okolo šesti procent, má tento operační systém do budoucna velký potenciál. Renomovaná zahraniční analytická společnost Gartner dokonce předpokládá, že se Microsoft stane na poli mobilních operačních systémů do roku 2015 druhým největším hráčem s podílem dosahujícím téměř dvaceti procent[3]. Lepší vyhlídky předpovídá Gartner pouze operačnímu systému Android společnosti Google (viz graf na obrázku 3.5). Do statistik Microsoftu je zahrnut i předchozí operační systém z dílny této společnosti – Windows Mobile, jehož další vývoj byl s nástupem WP zastaven. V blízké budoucnosti se dá očekávat i zastavení podpory tohoto operačního systému následované poklesem počtu jeho uživatelů.



Obrázek 3.5: Předpověď rozložení trhu mobilních OS společnosti Gartner

Velkou výhodou, kterou přináší volba této platformy, je představa programátora o vlastnostech koncového zařízení, na kterém bude vyvíjená aplikace provozována. Microsoft se rozhodl omezit přístroje s operačním systémem WP7 následujícími minimálními hardwarovými požadavky:

- display s rozlišením 800 x 480;
- kapacitní display schopný reakce na 4 doteky prstů najednou;
- DirectX 9 hardwarová akcelerace;
- integrovaný GPS přijímač;

[2]Obrázky převzaty z `http://mypidion.com/` a `http://www.rfid-in-china.com/`
[3]Statistiky převzaty z `http://www.gartner.com/it/page.jsp?id=1622614`

- akcelerometr;
- kompas;
- fotoaparát s bleskem;
- hardwarová tlačítka Zpět, Start a Hledat;
- podpora Wi-Fi;
- paměť RAM s minimální kapacitou 256 MB a FLASH paměť s kapacitou 8 GB.

Vývojářům díky těmto hardwarovým omezením odpadá starost o nejednotnost zobrazení aplikace na obrazovkách s různými rozlišeními. Pro potřeby této práce je také přínosem zaručená podpora Wi-Fi připojení, které, ačkoli je podporováno drtivou většinou v současnosti využívaných mobilních telefonů, nebylo doposud samozřejmostí.

Microsoft se rozhodl při příležitosti uvedení WP nevyvíjet nové frameworky ani implementační jazyky, ale využít již existujících nástrojů a technologií. Vývojáři si tak mohou zvolit mezi jazyky C# nebo Visual Basic .NET. Oba zmíněné jazyky umožňují programátorovi využití funkcí .NET Compact Frameworku. Aplikační platforma WP se dále skládá ze dvou dalších frameworků a to Silverlight a XNA. Jak Silverlight, tak XNA běží v aplikaci nad .NET Compact Frameworkem a od WP verze 7.5 je nově možné využití obou frameworků v rámci jedné aplikace[5]. Silverlight se primárně využívá pro byznys aplikace a jednoduché 2D hry a je využit i v této práci. Framework XNA je určen pro tvorbu pokročilých 2D a 3D her[3].

3.6 Konceptuální datové schéma

Ačkoliv zdrojem dat pro aplikaci je databáze uložená na centrálním terminálu a není tak třeba řešit způsob uložení dat v mobilním zařízení, je pro vytvoření tříd v aplikaci, které budou tato data reprezentovat, nanejvýš žádoucí znát alespoň elementární strukturu a vztahy mezi těmito daty. Pro zobrazení konceptuálního datového modelu (obrázek 3.6) byl použit Entity-relationship diagram. Diagram byl vytvořen na základě požadavků na aplikaci (viz 2.1) a obsahuje tedy pouze ta data, která jsou nezbytně nutná pro chod aplikace.

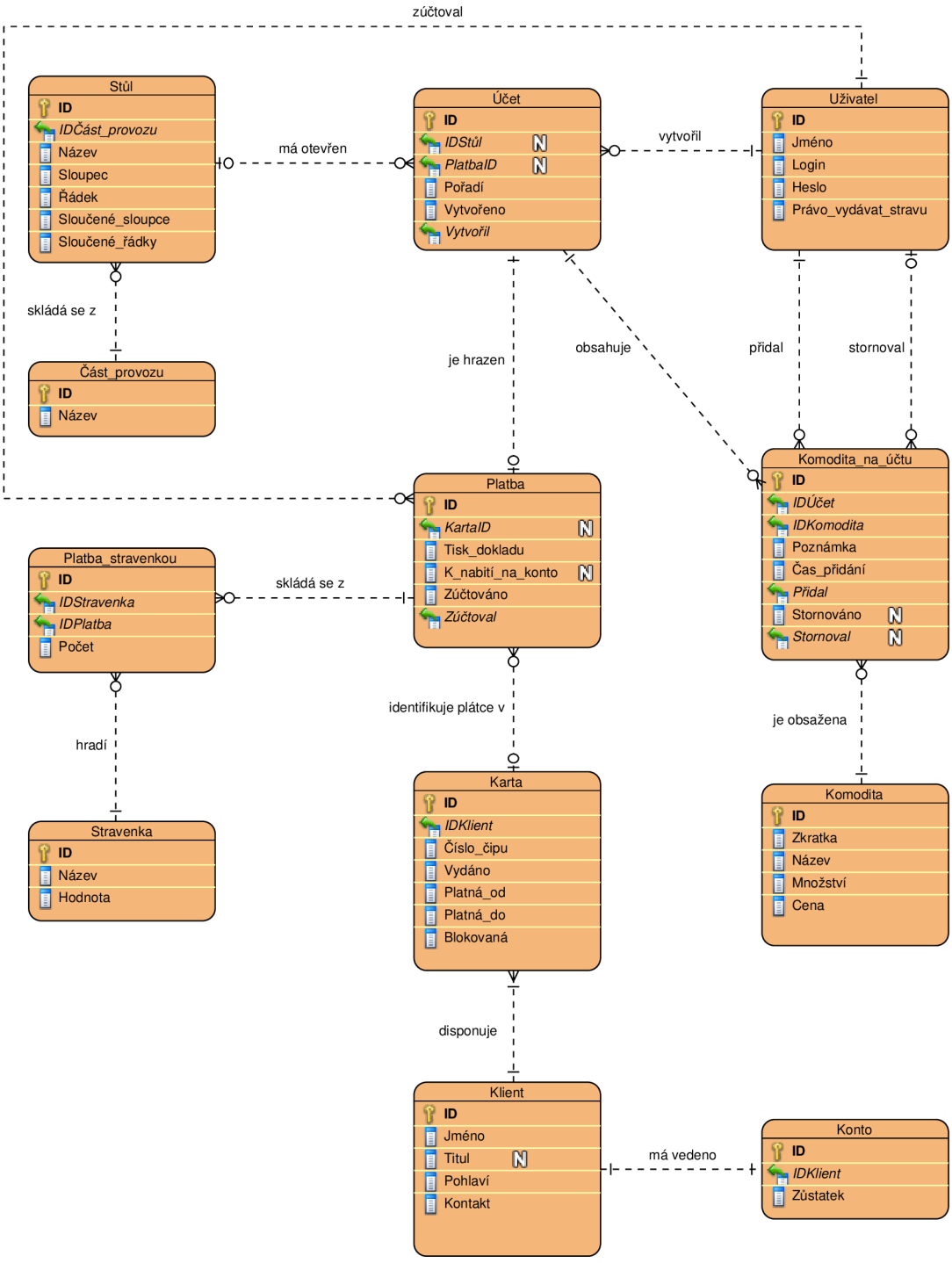
Při tvorbě diagramu jsem vycházel z předpokladu, že obsluhované prostory stravovacího provozu budou rozprostřeny na větší ploše. Z tohoto důvodu existuje entitní množina *Část_provozu*, která slouží k logickému oddělení jednotlivých částí provozu a umožňuje v implementační fázi jejich přehlednější zobrazení. Příkladem může být separátní zobrazení vnitřních prostor a zahrádky.

V jednotlivých částech provozu jsou rozmístěny stoly. Rozmístění stolů v aplikaci by mělo v rámci rychlé orientace obsluhy co nejvěrohodněji reflektovat skutečné rozmístění stolů, které se však bude provoz od provozu různit, proto by měly být informace o umístění stolu uloženy v databázi (podrobněji se tomuto tématu věnuji v podkapitole 4.2.1).

V průběhu provozní doby jsou na jednotlivých stolech vytvářeny účty. Účty by měly kromě jedinečného identifikátoru obsahovat atribut, který by pomohl obsluze od sebe odlišit účty např. v stiuaci, kdy je jich v jednu chvíli u stolu otevřeno několik a zároveň by byl snadněji zapamatovatelný než jedinečný identifikátor účtu. Touto informací by mohlo být ku příkladu pořadí vytvořeného účtu v aktuálním dni. Speciálním typem účtu bez vazby na stůl je účet vytvořen pro potřebu rychlého odběru.

Na účty jsou přidávány stravovacím zařízením nabízené komodity. Komodita by měla obsahovat minimálně informace o svém názvu a ceně, případně informaci o objemu nebo vázem komodity. Jelikož na účet smí být naúčtováno více komodit (a to i opakovaně) a jedna komodita se smí vyskytnout na více účtech, je informace o jednotlivých naúčtováních obsahem entitní množiny *Komodita_na_účtu*.

Při platbě nemusí být vždy placen celý účet, ale jeho platba může být rozdělena do několika dílčích plateb. V takovém případě je možností uložení platby v databázi několik. Já jsem v diagramu nastínil eventualitu, kdy při dělené platbě účtu, se pro odděleně hrazené komodity vytvoří dynamicky nový účet tak, byl vždy zachován vztah, že pro jeden účet existuje vždy maximálně jedna platba a jedna platba hradí právě jeden účet. Platba účtu může být provedena hotově, v takovém případě je identita klienta nepodstatná, nebo natížením útraty na konto klienta. Pro jednoduchost předpokládám, že každý klient má pouze jedno hlavní konto s informací o disponibilním zůstatku, které může být v případě potřeby dále děleno na dílčí konta. Klientovi může být přiřazeno více karet, ne všechny však musí být platné nebo aktivní.



Obrázek 3.6: Entity-relationship diagram[4]

[4]Pozn.: písmeno *N* u atributu značí, že daný atribut může nabývat hodnoty *null*

Kapitola 4

Implementace

Aplikace je implementovaná v programovacím jazyce Visual Basic .NET za použití WP SDK[1] 7.1 integrované do vývojového prostředí Visual Studio 2010. Podpora Visual Basic .NET je součástí SDK 7.1 na rozdíl od předchozí verze 7.0, kde bylo třeba tuto podporu doinstalovat. Cílovou verzí operačního systému (dále jen OS) WP je verze 7.1 uvedená na trh jako WP 7.5 s označením Mango. Aplikace je tak schopna využívat funkce a nástroje, jejichž podpora byla přidána až s uvedením OS 7.1. Daní za tuto volbu je nemožnost spustit aplikaci na předchozí verzi OS. V aplikaci byl při tvorbě uživatelského rozhraní použit rozšiřující balíček ovládacích prvků[2], které nejsou standartní součástí SDK.

4.1 MVVM

Ač význam návrhových vzorů roste s velikostí aplikace a výstupní aplikace této práce se svým rozsahem řadí mezi malé projekty, rozhodl jsem se v implementaci využít návrhového vzoru Model-View-ViewModel, zejména z důvodu lepší čitelnosti kódu a snazší implementace eventuálních oprav a rozšíření. Tento návrhový vzor je určen pro platformy, které k deklaraci uživatelského rozhraní využívají jazyk XAML (což použitá platforma Silverlight je).

Hlavním cílem MVVM je oprostit uživatelské rozhraní od aplikační logiky, která by měla být umístěna ve ViewModel třídách namísto tříd generovaných za každým souborem nesoucím deklaraci uživatelského rozhraní (tzv. code-behind). Díky tomu mohou být například při úpravách uživatelského rozhraní měněny názvy ovládacích prvků dle potřeby bez nežádoucího vlivu na chod aplikace. Kromě ViewModel tříd umístěných ve složce `\ViewModels` jsou dalšími součásti návrhového vzoru objekty typu View s definicí vlastního uživatelského rozhraní a objekty typu model reprezentující datové entity umístěné ve složkách `\Views` a `\Models`. Jednotlivé View objekty mohou *bindovat* vlastnosti obsažených ovládacích prvků na veřejné proměnné ViewModelu díky nastavení odpovídajícího ViewModelu ve vlastnosti `DataContext`. Nastavení se provádí následujícím způsobem:

```
<phone:PhoneApplicationPage.DataContext>
  <my:JidelnicekViewModel />
</phone:PhoneApplicationPage.DataContext>
```

[1] *Software Development Kit* – sada nástrojů pro vývojáře
[2] Dostupné na `http://silverlight.codeplex.com/releases/view/75888`

4.2 Uživatelské rozhraní

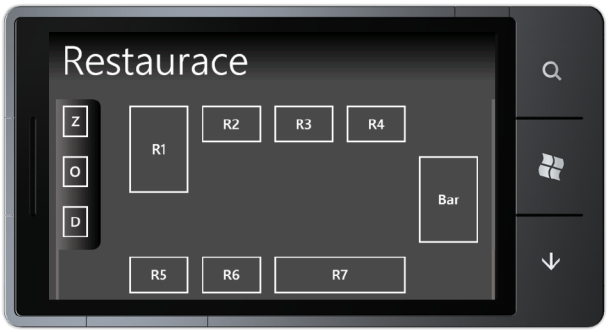
Každá jednotlivá obrazovka v aplikaci má svůj vzhled definován pomocí značkovacího jazyku XAML (z angl. Extensible Application Markup Language) a uložen v samostatném souboru. Jednou z předností jazyka XAML je možnost deklarovat vektorovou grafiku a animace, které jsou v aplikaci použity např. pro plynulý přechod vzhledu ovládacího prvku mezi dvěma vizuálními stavy. Deklarace jednoduché animace pro přechod ovládacího prvku ze stavu Neaktivní (*Unfocused*) do stavu Aktivní (*Focused*) vypadá v aplikaci takto:

```
...
<VisualTransition  From="Unfocused"
                   GeneratedDuration="0:0:0.3"
                   To="Focused">
     <Storyboard>
          <ColorAnimation Duration="0:0:0.3"
                          From="White"
                          To="Black"
                          Storyboard.TargetProperty="(Border.BorderBrush).
                                                     (SolidColorBrush.Color)"
                          Storyboard.TargetName="border" />
     </Storyboard>
</VisualTransition>
...
```

V této podkapitole se dále blíže zaměřím na vybrané dialogy z aplikace a moderní prvek uživatelského rozhraní ApplicationBar, který je použit napříč celou aplikací.

4.2.1 Dialog Provoz

Dialog se zobrazením stolů je specifický tím, že se bude jeho vzhled lišit na každém provozu, kde bude aplikace používaná. Jak už jsem se zmínil výše (podkapitola 3.6), je pro rychlou orientaci obsluhujícího personálu žádoucí, aby zobrazené rozestavení stolů v aplikaci co nejpřesněji reflektovalo skutečné umístění stolů. Předpokládám, že stoly jsou samostatná entitní množina, pro kterou je v databázi vyčleněna samostatná tabulka. Prvním krokem je



Obrázek 4.1: Dialog Provoz

tedy načtení stolů do kolekce. V aplikaci tuto kolekci reprezentuje třída `colStoly`, která obsahuje generickou kolekci `Stoly` typu `List(Of clsStul)`. Pro zobrazení kolekcí lze v XAML využít ovládací prvek `ItemsControl`, jemuž je do parametru `ItemsSource` předána generická kolekce stolů.

Samotné zobrazení stolů jsem se rozhodl řešit definováním mřížky s pevně danými rozměry. To, na jaké pozici v mřížce se stůl zobrazí, se rozhodne na základě informace o umístění stolu načtené z databáze (tato informace bude obsahovat souřadnice stolu a počet řádků a sloupců, do kterých se stůl vykreslí). Toto řešení umožní při každém spuštění aplikace načíst aktuální pozici stolů. Odpadá tak nutnost aktualizovat celou aplikaci v případě změny rozestavení stolů. Navíc toto řešení nabízí možnost implementovat rozhraní, přes které by si sami zákazníci mohli rozestavení stolů centrálně nastavovat a měnit. Náhled na dialog se zobrazením je zachycen na obrázku 4.1.

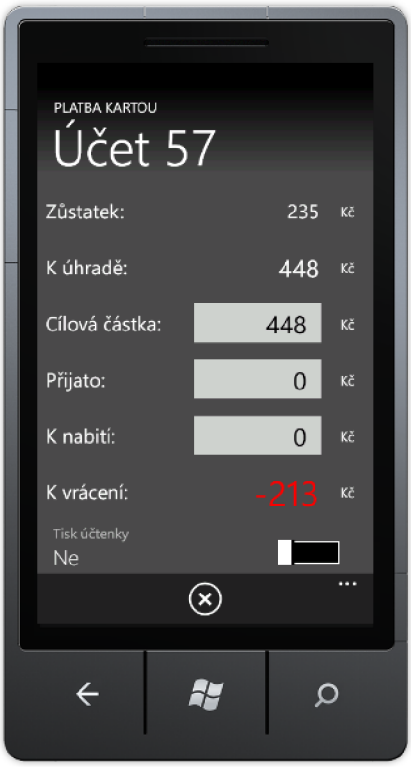
4.2.2 Dialog Načtení karty

Na obrázku 4.2 je zobrazeno uživatelské rozhraní dialogu pro načtení karty. Aplikace v této chvíli otevírá komunikační kanál se čtečkou a čeká na příchozí informaci o číslu čipu RFID karty. Aby bylo umožněno čerpat prostředky z kont i klientům, jejichž identifikační kartu se nepodaří načíst (důvodem může být např. mechanická závada na kartě), může obsluha zadat číslo karty ručně. Protože se čtečku RFID karet, na rozdíl od mobilního zařízení s WP, nepodařilo fyzicky zajistit, je v tomto dialogu tlačítko simulující načtení karty (pro potřeby aplikace se vygeneruje číslo čipu karty).

Obrázek 4.2: Dialog Načtení karty

4.2.3 Dialog Platba

Obrázek 4.3 zobrazuje uživatelské rozhraní dialogu pro platbu, v tomto případě platbu z konta. Tento dialog poskytuje obsluze informaci o disponibilním zůstatku na kontě klienta (Zůstatek) a o celkové hodnotě útraty (K úhradě), kterou zároveň dovoluje navýšit o spropitné (Cílová částka). Přijatá hotovost (Přijato) může být použita na uhrazení rozdílu mezi zůstatkem na kontě a hodnotou útraty, navýšení zůstatku na kontě nad rámec hodnoty objednávky nebo na kombinaci těchto akcí. V takovém případě musí obsluha určit, jaká část z přijaté hotovosti bude použita k nabití konta. Pro urychlení zadání částky určené k nabití na konto jsem implementoval popisek „K nabití“ jako přepínač, který po doteku mění nabíjenou částku na celou přijatou částku nebo rozdíl mezi hodnotou objednávky a stavem konta. Výše nabíjené částky může být zadaná také ručně pomocí softwarové klávesnice. Hodnota k vrácení vyjadřuje, kolik bude klientovi vráceno v hotovosti, nebo naopak kolik musí klient ještě doplatit (v takovém případě je zobrazena záporná hodnota nedoplatku červenou barvou). Dokončení platby je povoleno, pokud je částka k vrácení kladná nebo rovna 0. Dialog obsahuje přepínač, kterým lze centrální terminál informovat o tom, že k dané platbě si klient přeje vytisknout účtenku. Dialog pro platbu hotovostí je rozšířen o možnost zadat přijetí stravenek, naopak chybí informace o kontě a nabíjené částce.



Obrázek 4.3: Dialog Platba

4.2.4 ApplicationBar

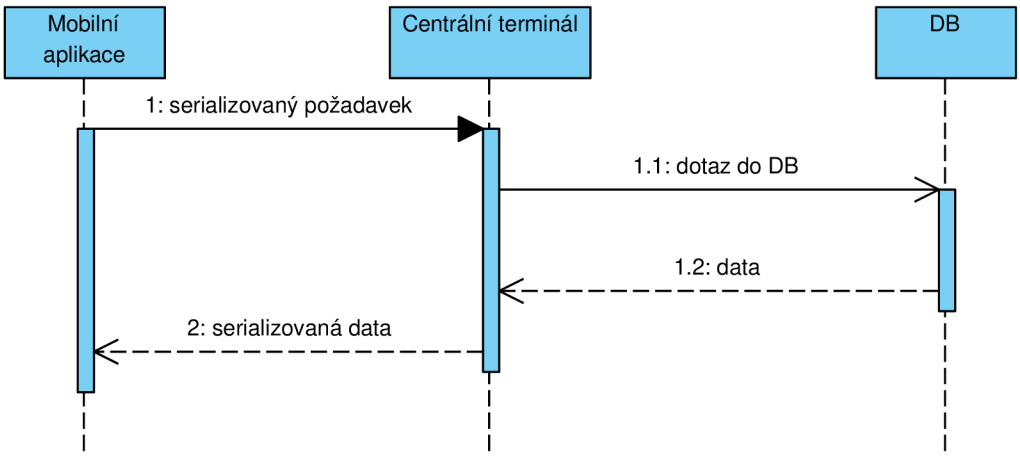
Tento ovládací prvek plní ve WP aplikacích podobnou funkci jako panel nástrojů v klasických desktopových aplikacích. ApplicationBar (zkráceně AppBar) a příbuzné třídy jsou defi-

novány ve jmenném prostoru `Microsoft.Phone.Shell` a nejsou součástí vizuálního stromu stránky. AppBar obsahuje menu položek složených z ikony a popisku (příklad na obrázku 4.4). Tato menu jsou použita na několika stránkách napříč celou aplikací. Při použití AppBar je třeba mít na paměti omezení, které s sebou tento moderní ovládací prvek nese. Prvním omezením je limit použitých položek v jednom menu. Maximální počet zobrazitelných tlačítek je omezen na 4. Dále není možné využít toto menu na stránkách typu *Panorama* (v aplikaci dialog Provoz). Je třeba si dát pozor na nestandardní chování při stisku tlačítka v menu, to totiž neodebere „focus“ jinému ovládacímu prvku na stránce. Jako příklad uvedu situaci, kdy mám na stránce ovládací prvek pro uživatelský vstup (např. `TextBox`) a tlačítko v menu pro potvrzení zadané hodnoty. Při stisknu tlačítka ve chvíli, kdy je prvek pro vstup stále aktivní, nebude potvrzen nově vložený obsah vstupního prvku, nýbrž obsah předcházející. Je zodpovědností programátora, mít tato omezení na zřeteli a postupovat při implementaci tak, aby nedocházelo k nežádoucímu chování aplikace.

Obrázek 4.4: Ukázka použití ApplicationBar menu

4.3 Komunikace se serverem

Komunikaci se stanicí, která zprostředkovává data pro mobilní terminály (v této kapitole ji budu označovat jako centrální terminál), obstarává třída `SocketClient`. Komunikace probíhá na transportní vrstvě (dle referenčního síťového modelu ISO/OSI) s využitím spojovaného protokolu TCP. Podpora komunikace přes sockety je jednou z nově přidaných funkcionalit ve WP 7.5. Třída obsahuje metody pro připojení k serveru, odeslání dat a přijetí odpovědi. Pokus o spojení se serverem je proveden asynchronně. Každý takový pokus je následovaný voláním metody `WaitOne()` objektu třídy `ManualResetEvent`. Toto volání způsobí pozastavení hlavního vlákna aplikace a zároveň spustí časovač po dobu, která je metodě předána jako parametr. Blokování hlavního vlákna může být ukončeno dvěma způ-



Obrázek 4.5: Proces získání dat

soby. Prvním případem je, když se aplikaci podaří spojit se serverem a obdrží odpověď, která ovšem nemusí nutně znamenat úspěšný průběh operace. Odpověď od serveru může signalizovat např. zamítnutí žádosti o komunikaci. Druhým způsobem ukončení blokování hlavního vlákna je vypršení zadaného časového limitu a aplikace pokračuje v hlavním vlákně s tím, že se spojení se serverem nepodařilo navázat.

Příprava dat k odeslání probíhá v modulu `RemoteDB` (v jazyce Visual Basic .Net jsou v modulu všechny metody a proměnné implicitně deklarovány jako statické a nelze vytvářet jeho instance). Data určená k odeslání jsou nejprve serializována pomocí instance třídy `XmlSerializer` a poté proudově odeslána, přijatý proud dat je pak pomocí téže instance deserializován. Celý proces získání dat je naznačen na obrázku 4.5.

4.4 Bezpečnost

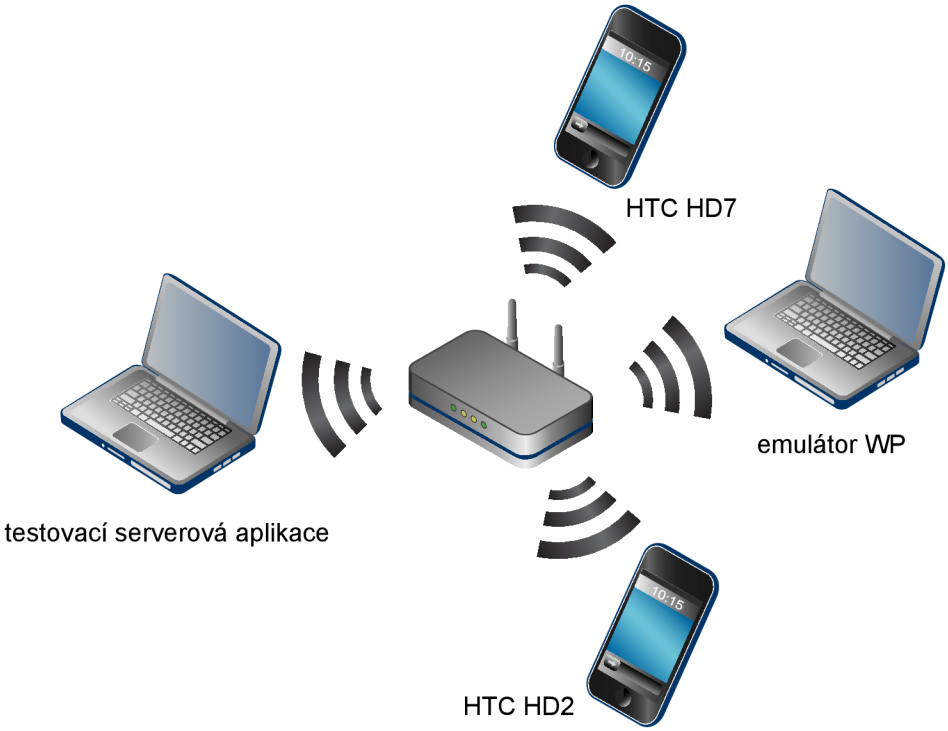
Jelikož implementovaná aplikace pracuje s financemi, je třeba se zamyslet nad otázkou bezpečnosti. Bezpečnost komunikace mezi centrálním a jednotlivými mobilními terminály závisí z velké části na infrastruktuře daného stravovacího zařízení. Ideálním řešením by byl stav, kdy by existovala speciálně vytvořená zabezpečená bezdrátová síť pouze pro potřeby komunikace terminálů. K minimalizaci riziku vniku do sítě nežádoucího zařízení mohou být využity filtry MAC adres nebo omezení přístupu k portu, přes který bude vedena komunikace pro vybrané IP adresy, případně rozsahy adresového prostoru. Zamezení přihlášení uživatele, který nemá práva k výdeji stravy, je věcí interní bezpečnostní politiky provozu. Mobilní terminál pouze obdrží informaci, zda má uživateli přihlášení povolit či nikoliv. Další bezpečnostní riziko vzniká ve chvíli, kdy personál odloží mobilní terminál s přihlášenou aplikací. Tím prakticky poskytne komukoliv, kdo by se terminálu zmocnil, provádět nežádoucí operace s účty, objednávkami nebo platbami. Pro tyto případy je v aplikaci definován časový interval. Pokud je aplikace nečinná déle, než je hodnota tohoto intervalu, je po uživateli znovu vyžádáno přihlášení. Na centrálním terminálu pro tyto případy bude možnost úplně zamezit přihlášení z odcizeného nebo ztraceného zařízení.

Kapitola 5

Testování systému

Mobilní aplikaci není možné otestovat bez funkční serverové části. Z toho důvodu jsem implementoval testovací konzolovu aplikaci, která zpracová požadavky z mobilních terminálů. Po zpracování požadavku odesílá na daný terminál relevantní odpověď. Testovací serverová aplikace se skládá z jednoho hlavního modulu, který implementuje celou komunikaci s klientskými aplikacemi a tříd reprezentujících perzistentní data. Testovací serverová část systému je ve výchozím stavu nastavená tak, že naslouchá požadavkům na všech síťových rozhraních stanice, na které je spuštěna. Výchozím portem je port číslo 8888 a přístupové jméno i heslo jsou nastaveny na „a“.

V průběhu vývoje jsem aplikaci testoval na emulátoru WP7, který poskytuje základní představu o tom, jak bude aplikace vypadat a fungovat na reálném zařízení. V dalších



Obrázek 5.1: Použitá zařízení a jejich propojení v poslední fázi testování

fázích jsem aplikaci nahrál na skutečná zařízení s operačním systémem Windows Phone 7.1 (konkrétně na mobilní telefony HTC HD7 a HTC HD2). Při těchto testech jsem se zaměřil především na aspekty, které nemohly být dostatečně otestovány v emulovaném prostředí. Na základě těchto testů jsem provedl několik úprav uživatelského rozhraní na místech, kde se špatná nebo zhoršená ovladatelnost aplikace při ovládání myší v emulátoru neprojevila.

V poslední fázi testů jsem zkoumal rychlost zpracování a přenosu dat po síti, délku prodlevy a stabilitu aplikace při jejím současném běhu na více zařízeních. Maximální dobu zpracování celého jednoho výpočetního cyklu (tzn. sestavení požadavku, odeslání na server, přijetí a zobrazení dat) jsem nastavil na 1 vteřinu. Pokud by provedení každého cyklu trvalo déle, zmenšovala by se časová úspora oproti klasickému papírovému zpracování objednávky a aplikace by ztrácela smysl. Při testech se ale ukázalo, že tento timeout byl dostatečný. Doba provedení výpočetních cyklů se pohybovala v intervalu od 16 do 970 ms, přičemž střední hodnota byla přibližně 420 ms. Nejdelší prodlevu vykazovaly samozřejmě cykly, ve kterých docházelo k přenosu většího množství dat. Tato situace nastává například při zobrazení jídelníčku, který obvykle obsahuje několik desítek položek a jehož načtení by bylo možné urychlit načtením statických položek jídelníčku do paměti mobilního zařízení při prvním spuštění aplikace. Tím by se omezilo množství přenášených dat pouze na dynamicky se měnící položky, jakými mohou být např. položky denního menu.

Aplikaci jsem testoval, jak znázorňuje obrázek 5.1, při současném běhu na třech zařízeních, přičemž nedošlo k patrnému prodloužení doby výpočetního cyklu. Data získává server z databáze jednoduchými, ve většině případů selektivními dotazy. Za předpokladu, že je serverová aplikace implementována tak, že každý nový požadavek je zpracováván v novém vlákně, nemělo by ani při větším počtu současně běžících zařízení docházet k výraznějším časovým prodlevám.

Kapitola 6

Závěr

Smyslem práce bylo navrhnout a implementovat na vhodně zvolené mobilní platformě aplikaci, která umožní v restauračních zařízeních vytváření a práci s objednávkami přímo od stolů. Aplikace byla implementována jako rozšíření k již existujícímu a používanému systému. Při vývoji byl kladen důraz na minimalizaci nůtných úkonů potřebných k začlenění aplikace do stávajícího systému. Motivací pro tuto práci bylo především nabídnout stravovacím podnikům efektivní možnost zúčtování objednávky a jejich klientům po identifikaci RFID kartou pohodlný způsob bezhotovostní platby z konta vedeného u daného podniku. Správa objednávek prostřednictvím mobilních terminálů přináší také časovou úsporu v podobě šetření času potřebného na předání objednávky k vyřízení. Po analýze požadavků na aplikaci a na její pozdější provoz byla zvolena cílová mobilní platforma Windows Phone, která poskytla dostatečné nástroje pro plnohodnotnou implementaci.

Mobilní aplikace komunikuje s centrálním systémem prostřednictvím lokální Wi-Fi sítě. Datová výměna probíhá serializováním dat do formátu XML a odeslání po síti prostřednictvím socketů. Uživateli, kterým bude s největší pravděpodobností personál restauračního zařízení, aplikace umožňuje elementární úkony, jako jsou vytvoření účtu, přesun účtu mezi stoly, hromadné nebo jednotlivé přidání komodit na účet a odeslání případných doplňujících informací spolu s objednávkou. Ve fázi platby aplikace umožňuje platbu v hotovosti či natížení útraty na klientovo konto po ověření klientovi identity identifikační kartou.

Funkčnost aplikace byla testována v emulátoru prostředí WP a na mobilních zařízeních HTC HD2 a HTC HD7. Rychlost odezvy aplikace se i při současném běhu na více zařízeních pohybovala pod 1 vteřinou se střední hodnotou pohybující se kolem 420 ms, což jsou hodnoty, při kterých je aplikace využitelná i v praxi.

Před nasazením aplikace do praxe by bylo nanejvýš vhodné nechat aplikaci otestovat dostatečným počtem kvalifikovaných osob (tzn. personálem stravovacích zařízení, pro který je aplikace primárně určena) a vyhodnotit požadavky na úpravy uživatelského rozhraní, které by vedly k rychlejší orientaci a snažší manipulaci s aplikací. Další potenciálně zajímavou úpravou by mohl být systém notifikace dokončení přípravy objednávky. V praxi by to mohlo vypadat například tak, že by se ve chvíli, kdy by byla objednávka připravená k podání zákazníkovi, odeslala na terminál ze kterého byla objednávka vytvořena informace o připravené objednávce. Ten by pak upozornil obsluhující personál např. vibračním signálem. Aplikace byla implementována s ohledem na snadnou udržovatelnost a bezproblémovou implementaci dalších funkcí, které by při komerčním nasazení byly po aplikaci případně vyžadovány.

Literatura

[1] Heartland Payment Systems. *Mobuyle,– Heartland Mobile Payments* [online]. 2011 [cit. 25. ledna 2012]. Dostupné na: <http://www.heartlandpaymentsystems.com/mobuyle/>.

[2] Lacko, L. *Programujeme mobilní aplikace ve Visual Studiu .NET*. Brno: Computer Press, 2004. ISBN 80-251-0176-2.

[3] Lee, H. a Chuvyrov, E. *Beginning Windows Phone 7 Development*. 2. vyd. New York: Apress, 2011. ISBN 978-1-4302-3596-5.

[4] Macková, K. *O bezkontaktní platby mají zájem tři čtvrtiny Čechů* [online]. 2011, Aktualizováno 5.října 01:45 2011 [cit. 31. ledna 2012]. Dostupné na: <http://ekonomika.eurozpravy.cz/ceska-republika/35311-o-bezkontaktni-platby-maji-zajem-tri-ctvrtiny-cechu/>.

[5] Microsoft. *Application Platform Overview for Windows Phone* [online]. 2012 [cit. 10. března 2012]. Dostupné na: <http://msdn.microsoft.com/en-us/library/ff402531(v=vs.92).aspx>.

[6] NXP Semiconductors. *MIFARE Smartcard IC's* [online]. [cit. 29. října 2011]. Dostupné na: <http://www.mifare.net/products/mifare-smartcard-ic-s/>.

[7] Smart Software. *SmartPOS* [online]. [cit. 25. ledna 2012]. Dostupné na: <http://smartpos.cz/>.

[8] Trilobite Technologies. *Meet•U – digitální dotykové terminály a kiosky* [online]. 2009 [cit. 25. ledna 2012]. Dostupné na: <http://www.meetu.cz/>.

Příloha A

Obsah DVD

- **\bp:** zdrojové soubory technické zprávy
- **\bppdf:** technická zpráva ve formátu `pdf`
- **\MobilniTerminal:** zdrojové soubory mobilní aplikace
- **\SocketServerCmd:** zdrojové soubory testovací serverové aplikace