VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

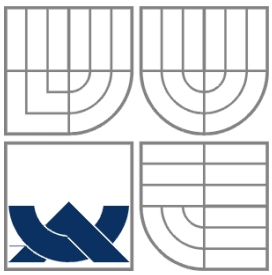
VYUŽITÍ PDA PRO DISTRIBUCI INFORMACÍ V RÁMCI UZAVŘENÝCH SÍTÍ

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

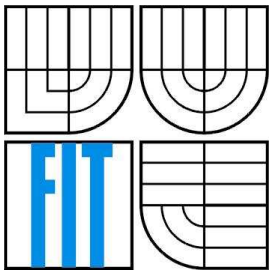
AUTOR PRÁCE
AUTHOR

BC. VOJTĚCH RYŠÁNEK

BRNO 2008



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

VYUŽITÍ PDA PRO DISTRIBUCI INFORMACÍ V RÁMCI UZAVŘENÝCH SÍTÍ

USAGE OF PDA FOR DISTRIBUTION OF INFORMATION IN CLOSED NET

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

BC. VOJTĚCH RYŠÁNEK

VEDOUCÍ PRÁCE
SUPERVISOR

ING. ROMAN LUKÁŠ, PH.D.

BRNO 2008

Abstrakt

Úkolem je naprogramovat aplikaci, která zpracovává multimediální data v rámci uzavřených sítí a tato data prezentovat uživateli. Modelově se tato problematika dá pojmout například jako průvodce městem – na základě pozice (GPS) podává uživateli patřičné informace o okolí. Aplikace musí umět do kapesního zařízení – PDA stahovat multimediální informace o okolních objektech a tyto informace prezentovat uživateli, pro kterého by zajisté nemělo být těžké tuto aplikaci ovládat.

Klíčová slova

Mobilní aplikace, PDA, GPS, Pocket PC, Microsoft Visual Studio, .NET, SQL server, webové služby, vlákna, ClickOnce, CLR, Garbage Collector, MSIL, JIT, Microsoft Windows, multimédia, html, byznys logika.

Abstract

The task of this project is to build application for managing of multimedia data in enclosed wireless networks and to present this data to the user. In model case it can be for example city guide – on position given by GPS are presented information about surrounding scene. Application must be capable of downloading data into Pocket PC – PDA and present them to the user as user friendly application.

Keywords

Mobile applications, PDA, GPS, Pocket PC, Microsoft Visual Studio, .NET, SQL server, Web services, thread, ClickOnce, CLR, Garbage Collector, MSIL, JIT, Microsoft Windows, multimedia, html, business logic.

Citace

Ryšánek Vojtěch: Využití PDA pro distribuci informací v rámci uzavřených sítí. Brno, 2008, diplomová práce, FIT VUT v Brně.

Využití PDA pro distribuci informací v rámci uzavřených sítí

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením Ing. Romana Lukáše, Ph.D.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Jméno Příjmení
Datum

Poděkování

Děkuji svému vedoucímu Ing. Romanu Lukášovi, Ph.D. za vedení mé diplomové práce a podporu při jejím vypracování.

Obsah

Seznam obrázků

1 Úvod

Problematika týkající se vývoje mobilních aplikací zahrnuje znalosti programování na obvyklých platformách s rozšířením o další rozměr v podobě menších výpočetních zdrojů a prostředků, které mohou být aplikaci poskytnuty. Při zahrnutí implementace distribuce dat, a to navíc multimediálních, je nutné mít tento pojem ještě více na paměti.

Zkratka PDA vznikla ze slov Personal Digital Assistant, což znamená osobní digitální pomocník, jak bylo toto zařízení původně koncipováno. Jedná se o malé zařízení s převážně velkou dotykovou obrazovku, která se dá ovládat pomocí pera (označuje se jako stylus). Současná PDA jsou poměrně velmi výkonná a zvládají přehrávat video, audio soubory a spouštět mnoho jiných náročných aplikací. Existuje několik operačních systémů pro tato zařízení, z nejznámějších můžeme uvést např. Windows CE, Windows Mobile (v současné verzi 6.0) a PalmOS, ale není také výjimkou spatřit zařízení s operačním systémem Linux.

V oblasti hardwaru těchto zařízení je k dispozici aktuálně procesor o kmitočtu něco málo přes 400 MHz a paměti RAM 64-128 MB. Kapacity pamětí se však momentálně rychle navyšují při současném poklesu jejich cen. Tato paměť se pak dělí na pracovní, kterou je možné využít pro uložení aplikací a na operační, která se používá pro samotný běh aplikací. Dále je k dispozici paměť typu ROM, ve které jsou uložena tzv. stálá data, např. operační systém a nastavení, které je nutné uchovat i po vypnutí přístroje. Paměť ROM, na rozdíl od předchozího typu RAM, není závislá na napájení, její obsah tedy zůstává i po odejmutí všech energetických zdrojů nezměněn.

K dispozici jsou také zpravidla sloty, nejčastěji pro paměťové karty, které jsou však využitelné i pro nepřeberné množství doplňkových zařízení, a to od GPS modulu (Global Positioning System – sytém, který umožňuje určit pozici uživatele tohoto zařízení na Zemi) přes fotoaparát, Wi-Fi (vysokorychlostní bezdrátové připojení) až k čtečce RFID čipů (Radio-Frequency Identification - identifikátor navržený k identifikaci zboží a jiných předmětů založený na bezdrátovém přístupu, který navazuje na systém čárových kódů). Nejčastěji se však tyto sloty používají pro paměťovou kartu k rozšíření kapacity pro uložení dat a programů. Existují rovněž verze PDA, které obsahují slot pro SIM kartu a dokáží rovněž kromě všech výše uvedených věcí fungovat i jako mobilní telefon, což je rovněž velký přínos pro uživatele. V současné době moderní PDA začínají obsahovat paměti typu flash (nevolatilni-semipermanentní paměť typu RAM s náhodným přístupem), čímž výrazně stoupá kapacita těchto zařízení a není tak výjimkou PDA s kapacitou disku 4GB a více.

Tato práce byla rozčleněna do sedmi kapitol vč. úvodu a závěru. V právě pročítaném úvodu, první kapitole, se čtenář seznamuje se základními informacemi vztahujícími se k danému problému.

Ve druhé kapitole bude zároveň uveden podrobnější popis vývoje mobilních aplikací jako takových, je zde představena historie vývoje na PDA a dále i vývoj současné moderní technologie .NET.

Ve třetí kapitole je probrán samotný pojem GPS, tzn. jakými způsoby a za pomocí jakých technologií je zjišťována pozice na Zemi.

Ve čtvrté kapitole je provedena analýza problému. Výsledkem analýzy jsou zde definovaná možná řešení problému, tj. stanovení souvisejících vlastností, cílových hledisek pro funkci a chování systému.

Pátá kapitola pojednává o implementaci problému podle předchozí analýzy. Je zde prezentováno vzniklé uživatelské rozhraní a funkčnost celého systému. U každé části systému jsou uvedeny nutné podmínky a nastavení pro jeho instalaci a správnou funkci, osvětleny jsou i technologie použité při implementaci.

Šestá kapitola pojednává o možném budoucím rozvoji aplikace, tzn. zahrnutí případných nových poznatků, informací a vylepšení korespondujících s eventuelními novými možnostmi systému.

V poslední části práce, závěru, se řešený problém uzavírá, problematika je na základě nových poznatků sumarizována spolu s dosaženými výsledky a zkušenostmi autora, které získal během vývoje této aplikace, přičemž nejsou samozřejmě opomenuty ani možnosti jejího dalšího uplatnění v praxi.

2 Vývoj pro mobilní aplikace

2.1 Historie

Kapesní zařízení mají poměrně všestranné použití, přičemž původně měla pomoci především s organizováním času a kontaktů, což už v současné době ne zcela platí. Obvykle obsahují programy pro uchovávání kontaktů, poznámek, textový a tabulkový editor, kalkulátor, přehrávač video, audio souborů a v neposlední řadě prohlížeč internetových stránek.

Vůbec prvními modely kapesních zařízení, pro které bylo možné vyvíjet aplikace, byly např. Hewlett Packard 200 LX a PSION 3A.

2.1.1 Hewlett Packard 200 LX

Přístroj si získal oblibu díky jeho kompatibilitě s tehdy rozšířeným operačním systémem MS-DOS. Byl postavený na bázi procesoru NEC, který byl kompatibilní s procesorem Intel x86. Obsahoval 1MB paměti RAM, která se dělila na mezipaměť pro operační systém a RAM-DISK. V paměti ROM byl uložený samotný operační systém, v tomto případě to byl o něco ochuzený MS-DOS ve verzi 3.0 a firemní software pro správu personálních údajů.

Přístroj měl sériový port, IrDA rozhranní a jeden slot pro PCMCIA kartu. S programovým vybavením pro tento model a ani s jeho vývojem nebyl žádný problém. Pokud tomu nebránila velikost paměti nebo rozlišení grafiky (CGA 640x200), fungovali na něm všechny programy pro MS-DOS včetně populárního SW Norton Commander a v našich končinách populárního textového procesoru T602. Protože HP 200 LX měl kompatibilní procesor s procesorem Intel 8086, bylo možné pro vývoj použít libovolný Assembler anebo vyšší programovací jazyk pro tento procesor na platformě MS-DOS. Po nainstalování jednoduchého interpretra programovacího jazyka BASIC bylo dokonce možné vytvořit aplikaci přímo na daném zařízení. Majitel paměťové karty si mohl nainstalovat i kompilátory vyšších programovacích jazyků, například jazyka C a vyvíjet aplikaci přímo v něm.

2.1.2 PSION série 3A

Měl vlastní operační systém, který obsahoval vyspělý firemní software pro správu personálních údajů. Do přístroje bylo možné zasunout dvě FLASH karty a byla k němu dodávána i lokalizace do češtiny a slovenštiny. Vývoj aplikací na platformě PC byl o něco složitější, naopak vývoj aplikací na kapesním počítači byl poměrně jednoduchý. Počítač měl vlastní operační systém s vestavěným programovacím jazykem OPL, což byl jazyk vzdáleně podobný BASICu. Kromě jazyka OPL bylo možné aplikace vyvíjet ve vývojovém prostředí SIBO pod operačním systémem MS-DOS. Tento

balík obsahoval výkonný kompilátor jazyka C nazývaný Speed C. Po nainstalování knihoven SIBO bylo možné vyvíjet kvalitní aplikace, které mohly využít všechny výhody multitaskingového operačního systému a vestavěných periferií. Co se týče kompatibility, společnost PSION přešla s novým modelem PSION série 5 na operační systém EPOC, bylo však nutné koupit nové vývojové prostředí pro tento operační systém. Toto prostředí se dodávalo jako doplněk k známému vývojovému prostředí Microsoft Visual Basic.

Windows CE přišlo s více nadějnější situací kompatibility u nového operačního systému Microsoft Windows CE ve verzích 1.0 a 2.0. Protože nebylo možné používat programy napsané pro Windows 95 či NT, ani programy pro MS-DOS, bylo nutné vyvíjet toto programové vybavení pro danou třídu počítačů samostatně. Situaci navíc komplikovala skutečnost, že se v této třídě používalo několik nekompatibilních procesorů (MIPS, SH3, SH4, ARM…). Naštěstí byla zachována alespoň částečná kompatibilita na úrovni zdrojových kódů. Microsoft pro vývoj na této platformě dodával softwarový balík Visual Basic a Visual C++.

2.2 Současnost

V současné době máme na výběr z několika vývojových nástrojů:

- Microsoft eMbedded Visual Tools 3.0,
- Microsoft eMbedded Visual Tools 4.0.,
- Microsoft Mobile Toolkit (MMIT),
- Smart Device Extensions (SDE).

Posledně jmenovaný nástroj se instaluje jako doplněk do vývojového prostředí Visual Studio .NET, přičemž od verze 2003 je tento balíček ve vývojovém prostředí již plně integrován. Rovněž byl integrován i toolkit ASP.NET Mobile Controls, který slouží k vývoji dynamických stránek pro zařízení PDA. V současné době je k dispozici Visual Studio .NET ve verzi 2005, které toto vše rovněž obsahuje a výrazně rozšiřuje možnosti tvorby aplikací všeho druhu.

2.2.1 Visual Studio .NET verze 2005/2008

S uvedením této nové verze se výrazně rozšířily možnosti vývojářů, kteří se pod tímto nástrojem rozhodli vyvíjet aplikace. Poprvé jsou totiž dostupné tyto nástroje ve formě speciální edice vývojářských produktů společnosti Microsoft zcela zdarma. Jednotlivé edice se dělí na

- Visual Studio Team Developer,
- Visual Studio Team Architect,
- Visual Studio Team Tester,
- Visual Studio Team Suite (obsahuje všechny výše uvedené),
- Visual Studio Team Foundation Server

- Visual Studio 2005/2008 Professional Edition,
- Visual Studio 2005/2008 Standart Edition.

Následně je pak volně k použití edice Express, která je zdarma a kdokoliv si ji může stáhnout z webu Microsoft. Jako jedny z nejvýraznějších změn oproti předchozí verzi tohoto vývojového nástroje jsou dále vysvětlené nové možnosti či technologie jako například refractoring, intellisense (s ním spojené automatické opravy), code snippets, technologie Click Once a mnohé další, přičemž ale především právě uveden se jeví z hlediska uživatele vývojového prostředí ve spojení s danou problematiku jako nejvýraznější.

Refractoringem se rozumí změna struktury a obsahu kódu zdrojového programu, tzn. pomocí tohoto nástroje je možné

- měnit názvy proměnných, procedur,
- extrahovat metody, tzn. označený kousek kódu je možné extrahovat jako novou metodu,
- provádět s kódem související operace.

Intellisense vytváří uživateli příjemné prostředí v podobě predikce příkazů a nabídky možností ke zvolení, tzn. nechá uživatele vybrat patřičnou funkci při psaní kódu, místo aby ji uživatel celou musel sám vypisovat. Vývojový nástroj se tímto pokouší odhadnout, co chce asi uživatel napsat. V nové verzi kontroluje např. i typy proměnných, tzn. dokáže nabídnout uživateli při zadávání argumentů metody ihned ten argument, který uživatel chce zadat. S tímto souvisí i code snippets, což je nabídka základních programových struktur, které vývojové prostředí vytvoří za uživatele a ten si jen upraví potřebné části kódu.

2.2.1.1 Technologie ClickOnce

Tato technologie je určená k nasazení a aktualizaci vyvíjených aplikací. Při implementaci této práce byla rovněž využita. Technologie ClickOnce umožňuje poměrně jednoduše a efektivně odstranit typické problémy nasazení aplikace jako např.

- proces nasazení aplikace,
- automatizace případných aktualizací,
- časté konflikty DLL knihoven,
- zefektivnění práce uživatele při instalaci.

ClickOnce umožňuje řadu efektivních kroků, některé z nich jsou na ukázku popsány v následujícím textu. Např. lze uvést možnost volby spouštění aplikace z webu nebo její možné nainstalování pro off-line použití. V případě off-line použití (tedy instalace) je možné nastavit způsob aktualizací, k dispozici jsou volby pro kontrolu aktualizací před spuštěním samotné aplikace. V takovém případě dojde k eventuální aktualizaci okamžitě, nebo až za běhu aplikace s tím, že se případný update provede před příštím spuštěním. Také je možné využít ClickOnce API a přidat položku pro manuální kontrolu aktualizací např. do menu. Další přínosnou možností je nastavení

minimálního čísla verze. Pokud pak aplikace zjistí, že je na webu aktualizace s vyšší zadanou minimální verzí, nebude již možné starou verzi použít, aniž by uživatel povolil její update. Vývojář tak vlastně může donutit uživatele k instalaci různých kritických oprav.

Další skupina nastavení se týká bezpečnosti. Aplikace postavená na .NET Frameworku standardně běží v tzv. sandboxu, což je obdoba kontejneru, ve kterém se spouštějí applety v rámci webového prohlížeče (tzn. neinstaluje se do obvyklých složek Program Fines apod.). Obvykle ale aplikace vyžaduje další práva, např. pro přístup k registrům nebo souborovému systému. V nastavení je možné zadat minimální práva, která aplikace potřebuje pro svůj běh. Při spuštění je pak uživatel dotázán, zda chce aplikaci tato práva přidělit, a nebo ji ukončit. Tím se zamezí jejímu selhání z důvodu odmítnutí přístupu k nutným systémovým zdrojům. Minimální práva je možné ve Visual Studiu nastavit buď ručně, a nebo je nechat vygenerovat automaticky pomocí tzv. kalkulátoru oprávnění. Dále je možné k aplikaci přidat různé bezpečnostní certifikáty, které umožňují ověřit její pravost a navíc zajistí, že se budou stahovat aktualizace pouze od původního vydavatele.

Po provedení těchto nastavení, a eventuelně pro vlastní implementaci ClickOnce API v programovém kódu, se pomocí nástroje publish wizard aplikace se všemi nutnými součástmi nakopíruje na web, případně máme možnost instalaci umístit na CD nebo na disk. Odtud si ji uživatel může přímo spustit nebo nainstalovat, případně se vytvoří CD s automatickým spuštěním instalace po vložení do mechaniky. Touto možností se dostáváme k poslednímu podstatnému prvku technologie ClickOnce. Uživatel nestahuje z webu přímo danou aplikaci, ale tzv. bootstrapper. Jedná se o program, jehož účelem je zajistit aplikaci veškerý potřebný software. Po stažení a spuštění bootstrapper nejprve zkontroluje, zda je na počítači nainstalován .NET Framework patřičné verze. Pokud ne, automaticky se postará o jeho stažení a instalaci (případně tohle vše přidá k instalaci na CD a uživatel nemusí nic stahovat). Stejným způsobem jsou obstarány všechny další softwarové součásti, které aplikace potřebuje, např. Microsoft DirectX nebo Microsoft SQL Server Express Edition apod. Pokud je to potřeba, postará se bootstrapper během tohoto procesu i o restarty systému. Teprve když je připraveno softwarové prostředí, přejde bootstrapper k samotnému spuštění nebo instalaci původní aplikace. Bohužel u online spuštění nebo instalace z webu je zatím podporován pouze prohlížeč Internet Explorer společnosti Microsoft, nicméně na možnosti toto provádět i v dalších oblíbených prohlížečích se pracuje.

2.2.2 Novinky ve Visual Studio .NET verze 2008

Ve verzi Visual Studia 2008 došlo k několika vylepšením, některá z nich jsou nastíněna v dalším textu. Hlavní změnou je plná podpora .NET frameworku ve verzi 3.5, jehož jednou z největších změn je zavedení LINQ, který je popsán dále.

Podpora Multi-Targetingu – studio umožňuje vyvíjet aplikace určené pro různé verze .NET frameworku, nejen čistě pro verzi 3.5. Tzn. je možné vyvíjet aplikace a upravovat aplikace pro ASP.NET verze 2.0 a pokračovat v jejich nasazení na .NET 2.0 serverech.

Vylepšená podpora AJAX a Javascriptu - .NET 3.5 již obsahuje zabudovanou podporu pro AJAX. Javascript rovněž má zahrnutou podporu Intellisence a je možné ho debugovat přímo za běhu aplikace, což dříve nebylo možné.

Silverlight technologie – v souvislosti s vylepšenou prací s Javascriptem byla zavedena nová technologie Silverlight, která umožňuje tvorbu aplikací podobně jak je to možné např. v technologii Flash. Technologie Silverlight je ale založena přímo na .NET architektuře, proto se na něm mohou stavět aplikace přímo založené např. na jazyku C#.

Vylepšená podpora návrhu webu – Web Designer a podpora CSS – došlo k výraznému vylepšení návrhu webových aplikací jak přes samotné rozvržení stránek, tak i přes CSS.

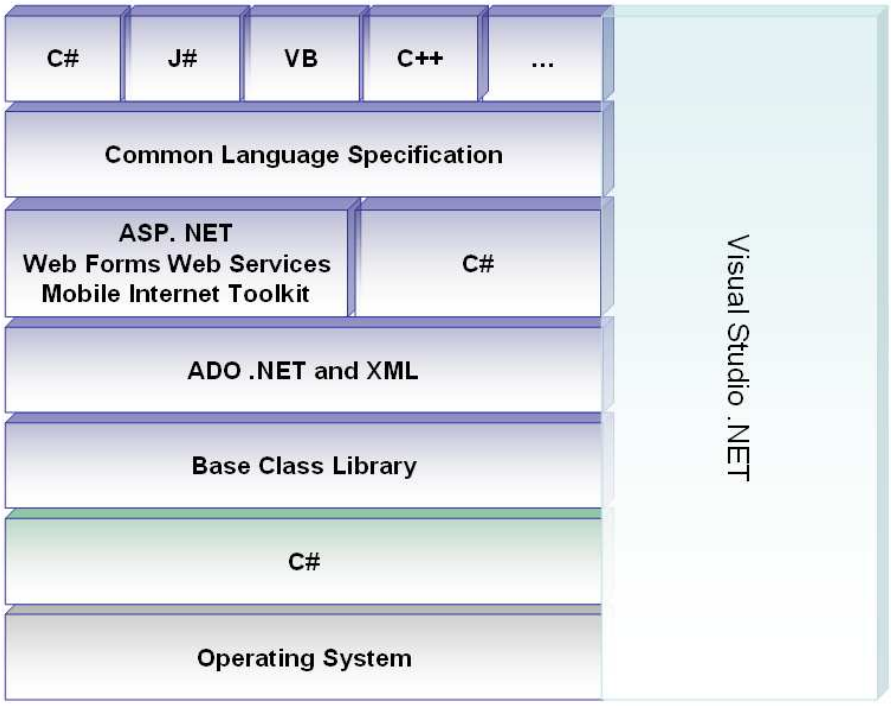
Technologie ClickOnce je již dostupná i v prohlížeči Firefox, což dříve nebyla.

2.2.2.1 LINQ

Jedna z největších inovací nového .NET frameworku. Jedná se ve zkratce o Language Integrated Query, tedy dotazovací jazyk integrovaný přímo v C# (nebo VB.NET). Tato funkčnost umožňuje programovat flexibilnější aplikace s čistějším kódem a umožňuje provádět dotazy podobné těm, které známe z SQL přímo v rámci zdrojových kódů naší aplikace na kolekcích, polích apod. Existují jeho dvě alternativy – LINQ to SQL a LINQ to XML a jak už název napovídá, jedná se o dotazování nad databázemi a XML soubory.

2.2.3 .NET architektura

Vývojové prostředí Visual Studio .NET zastřešuje vývoj aplikací jak pro desktop, tak pro mobilní zařízení, tzn. pomocí tohoto jednoho vývojového prostředí, samozřejmě s respektováním rozdílů mezi .NET Frameworkem pro desktop aplikace a .NET Compact Frameworkem pro mobilní zařízení, je možné vyvíjet aplikace jednak pro normální PC, tak i pro mobilní zařízení. Hlavní výhodou vývoje pod tímto nástrojem je vývoj kódu přenositelného mezi různými platformami, tzn. pro vývojáře odpadá nepříjemná práce s konverzí kódu pro různé platformy. Obrázek 2.1 znázorňuje současnou architekturu .NET. Jak je vidět, nahoře jsou všechny možné druhy jazyků, s kterými je možné pod touto platformou programovat, tzn. že např. programátor zvyklý na Visual Basic se nemusí nic výrazného učit v syntaxi jazyka a rovnou může začít s minimálními změnami programovat pod .NET.



Obrázek 2.1 – Architektura .NET Frameworku

Do této doby bylo nutné vyvíjet aplikace pro různé platformy zvlášť. Pokud byla provedena kompilace např. pod Delphi, pak výsledná aplikace byla zkompilována pro danou platformu, přičemž asi nejčastěji to bylo pro platformu Win32 operačních systémů Microsoft Windows, ale mohly to samozřejmě být i jiné. Znamená to, že zdrojový kód aplikace byl kompilací převeden do strojového kódu počítače. To ve výsledku přináší velmi dobrou rychlost běhu výsledné aplikace, avšak na druhou stranu z toho plynou i některé nevýhody – nepřenositelnost aplikace mezi jednotlivými platformami, popřípadě verzemi operačních systémů a nezřídka jsou k vidění chyby v přístupech do operační paměti.

Princip řízených běhových prostředí, použitý právě u platformy .NET (ale už i dříve u platformy Java společnosti Sun Microsystems), řeší tento problém poněkud trochu jinak – přidává k převodu zdrojového kódu do kódu strojového ještě jednu vrstvu. Tuto vrstvu představuje mezikód, do kterého jsou zdrojové kódy zkompilovány. Mezikód je pak běhovým prostředím na cílové platformě (Windows, Linux) převeden do strojového kódu.

Tento převod je na cílové platformě realizován vždy při spouštění dané aplikace. Vyšší náročnost na výkon uživatelského počítače je nepodstatným mínusem tohoto překladu a v dnešní době rychlých počítačů neznamená tedy žádné velké negativum. Navíc je nutno poznamenat, že při spuštění aplikace nedochází k překladu celé aplikace najednou, ale používá se tzv. JIT (Just-in-Time) kompilace. JIT kompilace znamená, že do strojového kódu je převedena pouze potřebná část mezikódu a při opětovném použití této (již přeložené) části se spouští její zkompilovaná forma, což se

příznivě projeví na rychlosti, která si již nezadá s během takto neřízeného programu. Aplikace postupně překládá kód, který je v dané chvíli nezbytný k jejímu dalšímu běhu.

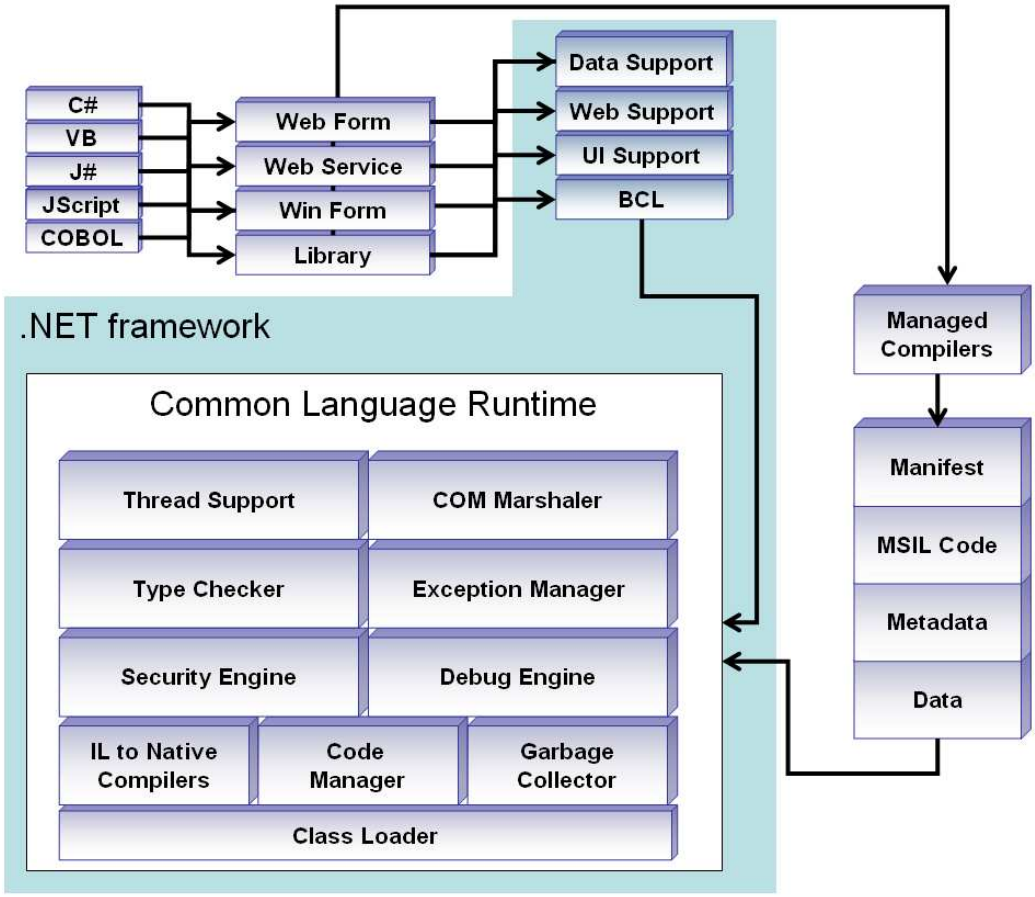
Microsoft tuto technologii v .NET frameworku nazval MSIL, tedy Microsoft Intermediate Language. Tento jazyk relativních adres je spouštěn klíčovou součástí .NET frameworku pojmenovanou CLR (Common Language Runtime neboli společné běhové prostředí) a společnost Microsoft jej dala ke standardizaci organizaci ECMA.

V CLR existuje komponenta pro práci s operační pamětí, která programátorovi velmi usnadňuje práci, tzv. Garbage Collector. Jak už název napovídá – doslovně „sběrač odpadků“, jedná se o sadu algoritmů pro uvolňování nepotřebných programových objektů z paměti. Díky Garbage Collectoru se již vývojáři nemusejí starat o přiřazování nebo uvolňování operační paměti a odpadá tak riziko nekorektní práce s ní, která dříve v mnoha situacích končila „pádem“ aplikace a někdy i blokací celého systému.

Další důležitou vlastností .NET frameworku je CLS – Common Language Specification, tedy společná jazyková specifikace a s mé související CTS – Common Type Systém, či-li společný typový systém. Výsledkem použití CLS a CTS je rovnocennost programovacích jazyků. Jinými slovy – pro vývoj .NET aplikací je možné použít jeden z několika programovacích jazyků vyšší úrovně, tak jak již bylo naznačeno v úvodu. Může se jednat například o:

- C#, nový jazyk vyvinutý pro .NET,
- Visual Basic .NET, nová generace oblíbeného jazyka Visual Basic,
- J#, což je jazyk se syntaxí rozšířeného jazyka Java,
- managed C++, kde slovíčko managed označuje možnost psát řízený kód pro .NET dokonce i v tak od svého počátku „neřízeném“ jazyce.

Obrázek 2.2 ukazuje právě detail této architektury, na které je dobře vidět propojení několika rozdílných jazyků do jedné architektury .NET frameworku.



Obrázek 2.2 – Detail architektury .NET frameworku

S tím souvisí i další výhoda této platformy – výrobcům třetích stran nic nebrání ve vývoji dalších jazyků. Jediné, co je nutné dodržet, aby tento nový jazyk měl kompilátor se schopností kompilovat zdrojové kódy do mezijazyku MSIL. To znamená splnit specifikaci CLS danou společností Microsoft. Právě na tuto možnost navázal projekt Mono, což je obdoba .NET frameworku v Linuxu, kdy většina aplikací naprogramovaných na Windows má schopnost běžet beze změn na platformě Linux, což je výrazný krok kupředu ve vývoji aplikací pro různé platformy. Nicméně Mono je zatím pouze v testovací verzi a zahrnuje v současné chvíli pouze verzi frameworku 1.1, ale na dalších verzích se pracuje a projekt vypadá velice slibně.

2.2.3.1 Tlustí a tencí klienti

Tlustý klient

V architektuře klient/server běží programy (přesněji aplikační logika a business logika) na dobře vybavených tzv. „tlustých grafických klientech“ a s daty uloženými na serverech v centrálních databázích. Na přelomu tisíciletí se v této oblasti začala prosazovat centralizovaná koncepce s označením vícevrstvá architektura.

Výhody vícevrstvé architektury

Ve vícevrstvé architektuře program běží na centrálních systémech vytvořených ze vzájemně spolupracujících aplikačních serverů. Na klientských stanicích se pouze zobrazují data a ovládací prvky pro styk uživatele s aplikací. Základní výhodou vícevrstvé architektury při údržbě programu je proto snadná údržba aplikací a jejich datových zdrojů daná již zmíněnou centralizací. V případě (nového) tenkého klienta se totiž kromě výměny části centrální aplikace na aplikačním serveru čas od času provede – plně automaticky a pro uživatele skrytě – jeho jednoduchá aktualizace (update) po síti. Opravy chyb, změny verzí a drobná vylepšení programů se již nemusejí složitě provádět na tisících počítačů, stačí zde pouhá výměna příslušné části centrální aplikace na aplikačních serverech. Vzhledem k centralizaci tedy odpadají problémy spojené s distribuovaným zpracováním dat, jako je sehrávání datových souborů s následnou synchronizací a ztotožňováním. S podporou nového technického vybavení (clustery, externí disková pole, grid-technologie a blade-servery) se snadno řeší i problémy svázané se škálovatelností a se zálohováním informačního systému.

Tenký klient

Uživateli se údaje často zobrazují na koncovém terminálu (stanici), v univerzálním a snadno dostupném internetovém prohlížeči (HTML), který nemá přílišné nároky na zdroje. Taková klientská stanice se proto někdy také označuje jako „tenký klient“ HTML. Současný vývoj však ukazuje, že snížení komfortu ovládání aplikací, které je vynucené omezenými možnostmi jazyka HTML, je příliš znatelné.

Začíná se proto čím dál více objevovat komfortnější koncové prostředí. Jedná se o klientské programy využívající plně grafické možnosti operačních systémů k tomu, aby uživateli nabídly všechen ovládací komfort, na který je zvyklý ze starších systémů klient/server. Aby se odlišili „noví“ klienti z vícevrstvé architektury od „starých“ klientů z architektury klient/server, zavedl se ve vícevrstvé architektuře také pojem tenký klient. Tento druh terminálových programů lze vytvářet jak v prostředí .NET společnosti Microsoft, tak i jako aplikace nebo applety v jazyce Java s využitím grafických knihoven Swing.

3 GPS – Global Positioning System

3.1 Úvod

Zkratkou GPS je myšlen navigační systém (původně vojenský), který dokáže s několikametrovou přesností určit pozici kdekoliv na Zemi. Toto určení polohy lze ještě zpřesnit použitím metod jako je např. diferenciální GPS (DGPS). Mimo polohy je možné přes GPS zjistit rychlost a směr, jakým se přijímač pohybuje a je také možné zjistit aktuální čas. Alternativy k GPS systému jsou ruský GLONASS a evropský Galileo.

3.2 Historie

Vývoj GPS sahá až do roku 1973, kdy se na něm začalo pracovat jakožto na vojenském projektu ministerstva obrany USA (původní označení znělo NAVSTAR GPS). Po roce 1983, kdy ruská stíhačka sestřelila ve vzdušném prostoru SSSR civilní korejské letadlo, oznámil prezident USA Ronald Reagan, že po dokončení projektu bude GPS dostupné i pro civilní použití. K tomu došlo v roce 1994, kdy byla na orbitu umístěna kompletní sestava všech 24 satelitů. Nicméně až do roku 2000 byla do měření pro civilní použití zanášena umělá chyba (označovaná jako SA – Selective Availability), která měla zabraňovat např. zneužití systému různými teroristickými skupinami pro navádění raket a jiné nežádoucí aktivity.

3.3 Určení polohy

Typický GPS přijímač určuje polohu za pomocí signálu ze čtyř a více satelitů. Čtyři satelity jsou potřeba kvůli přesnému určení lokálního času (přesnějšímu než dokáží získat normální hodiny). Jinými slovy přijímač používá tři proměnné k výpočtu – x, y, z a t. Tyto hodnoty jsou pak přepočteny do uživatelsky příhodné formy – zeměpisné šířky, výšky alias polohy na Zemi a jsou zobrazeny uživateli.

Každý satelit má v sobě atomické hodiny a neustále vysílá zprávy obsahující čas v době vyslání zprávy, parametry potřebné k výpočtu pozice satelitu. Signál ze satelitu cestuje známou rychlostí (rychlostí světla, zpomaleného průchodem přes atmosféru Země). Přijímač pak použije čas příchodu zprávy k výpočtu vzdálenosti satelitu a následně tak zjistí svoji polohu za použití geometrie a trigonometrie. Pokud by bylo určení místního času dostatečně přesné, tzn. časy na přijímači a na vysílači by byly synchronní, stačily by k určení polohy pouze 3 satelity. Nicméně k tomuto by bylo potřeba atomových hodin i v přijímači, což většina přijímačů neobsahuje. Pokud by totiž došlo k odchýlení času o jednu milisekundu, tak po násobení rychlostí světla by to byla vzdálenost 300 km,

což dělá výsledek měření nepoužitelným. Tímto vzniká tedy kromě tří-rozměrné polohy i další neznámá a tou je hodnota odchylky času na přijímači. Proto jsou potřeba 4 satelity k přesnému určení polohy. Většina přijímačů ale začne určovat polohu již při signálu ze tří satelitů a to proto, že se většina uživatelů nachází na zemském povrchu a proto je jejich poloha jen dvourozměrná. Přijímač s tímto faktem většinou počítá, a určí tak jen zeměpisnou šířku, délku a odchylku hodin. Nicméně protože povrch Země není zcela přesně definován (nemá jednotný jednoduchý tvar), tak takové určení polohy není dostatečně přesné a obsahuje poměrně velkou chybu.

Celkem kolem Země obíhá 24 satelitů, ze kterých jsou 3 záložní. Tyto satelity obíhají nad povrchem Země v šesti drahách ve výšce 20 200 km. V každém satelitu jsou tři až čtyři atomové hodiny a dále pak detektory kontrolující dodržování zákazu provádění zkoušek nukleárních zbraní. V našich zeměpisných šířkách je vidět průměrně 8 satelitů, nicméně je třeba počítat se zakřivením okolního terénu (ve městě signál mohou rušit budovy), což může počet využitelných družic zásadním způsobem snížit.

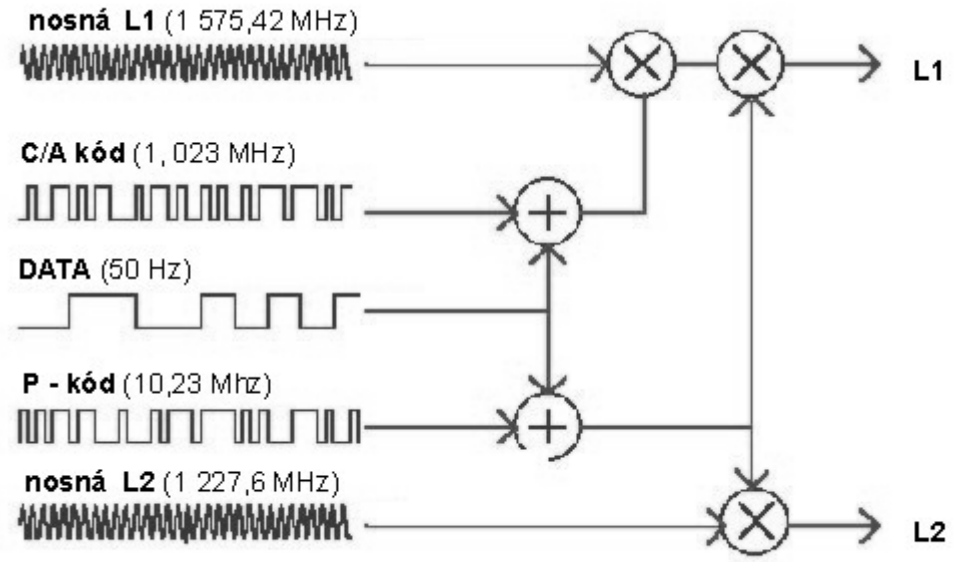
Výrazně vyšší přesnosti lze dosáhnout porovnáním naměřených hodnot s hodnotami naměřenými referenčním pozemským přijímačem (tzv. diferenciální GPS, DGPS). Tímto způsobem je možné provádět přesná geodetická měření s přesností na milimetry až centimetry. Takzvaná korekční doplňková data jsou přenášena jiným kanálem, například pomocí jiných satelitů či RDS (FM vlny), nebo na dlouhých vlnách (AM vlny). Přijímač korekčních dat je buď integrovaný v GPS nebo se připojí externě.

3.3.1 Technologie satelitů

Družice vysílají na několika kmitočtech, které jsou zvoleny záměrně, aby co nejlépe odolaly atmosférickým vlivům.

- **„L1“ (1575,42 MHz)**, kde je vysílán C/A kód (dostupná pro civilní uživatele systému).
- **„L2“ (1227,62 MHz)**, kde je šířen vojenský P/Y kód, který je šifrovaný (je přístupná pouze pro tzv. autorizované uživatele).
- **„L3“ (1381,05 MHz)** obsahuje signály, které souvisí s další funkcí systému GPS, odhalováním startů balistických raket (doplňuje tak satelity náležící k Defense Support Program), jaderných výbuchů a dalších vysokoenergetických zdrojů infračerveného záření.
- **„L4“ (1841,40 MHz)** se využívá pro měření ionosferického zpoždění. Průchod signálu ionosférou způsobuje totiž přidání dodatečného zpoždění ke zpoždění způsobenému vzdáleností, které se promítne do chyby polohy (viz následující podkapitola). Toto ionosférické zpoždění lze eliminovat měřením zpoždění na dvou kmitočtech.

- **„L5“ (1176,45 MHz)** se plánuje jako civilní safety-of-life (SoL) signál. Tato frekvence spadá do mezinárodně chráněné oblasti letecké navigace, ve které je malé nebo žádné rušení za všech podmínek. S vypuštěním prvního Block IIF satelitu, který bude poskytovat tento signál, se počítalo na rok 2007, v současné době by tedy měl být již v provozu.



Obrázek 3.1 – Vznik signálu GPS

3.3.1.1 Formát dat

Každá družice vysílá tok binárních dat, která jsou dělena do slov o 30 bitech. Ze 30 bitů je jen 24 informačních, ostatních 6 bitů slouží k zabezpečení přenosu K zabezpečení je použit Hammingův kód (32, 26) se vzdáleností 4. Deset slov tvoří podrámec a pět podrámců tvoří rámec. Protože jeden bit trvá 20 ms, je slovo dlouhé 0,6 s, podrámec 6 s a rámec 30 s.

První, druhý a třetí podrámec obsahuje aktuální informace o stavu družice vysílající danou navigační zprávu, jejich obsah se aktualizuje několikrát za den. Mezi okamžiky aktualizace je obsah podrámců konstantní. Čtvrtý a pátý podrámec obsahuje informace o celém systému GPS a jejich obsah se aktualizuje několikrát za týden. Mezi okamžiky aktualizace se obsah podrámců pravidelně opakuje s periodou 25 rámců.

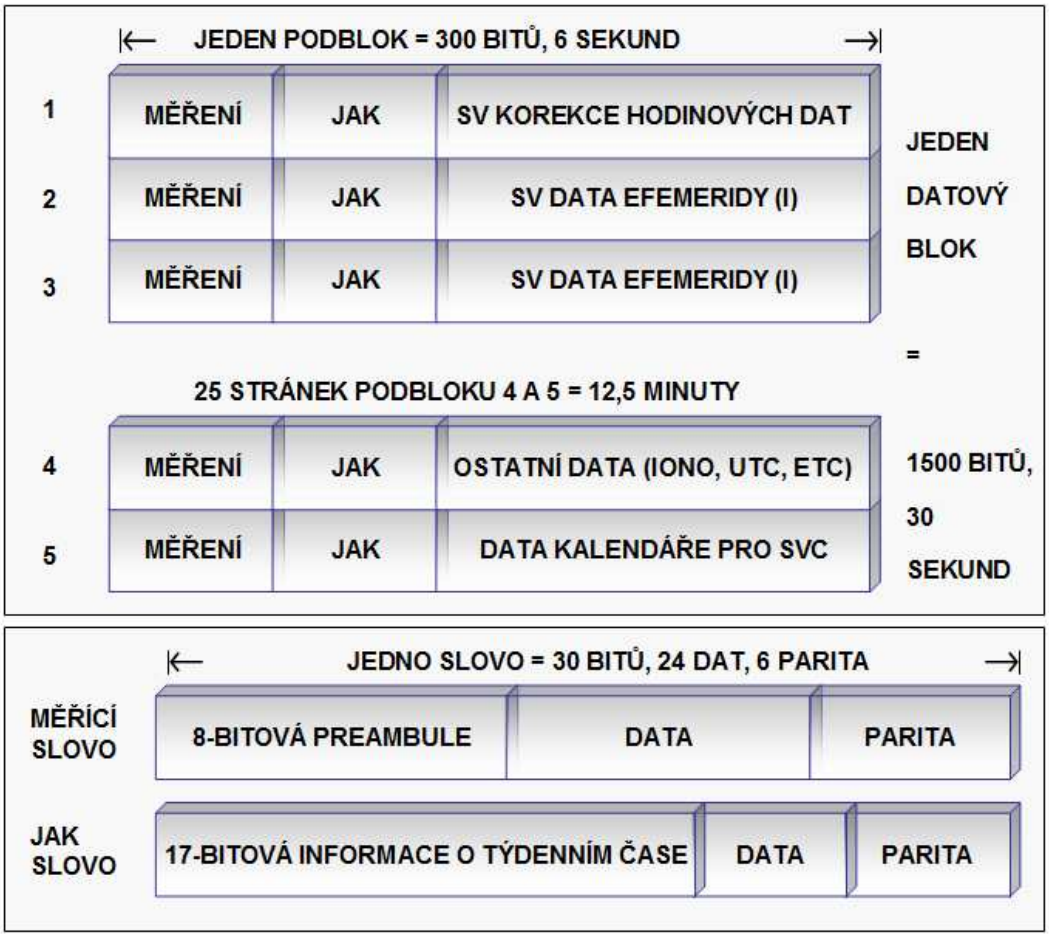
Celá informace o systému GPS je tedy obsažena v posloupnosti 5 * 25 = 125 podrámců. Této posloupnosti říkáme navigační zpráva. Podrámec daného čísla (1 až 5) má 25 možných významů, kterým říkáme stránka a označujeme je pořadovým číslem rámce ve zprávě. První stránka od začátku zprávy má číslo 1.

Pro přenos bitového toku se používá modulace BPSK (binary phase shift keying), tj. podle hodnoty bitu se fáze nosné mění o 180 stupňů. Protože všechny družice vysílají na stejném kmitočtu, je nutné jejich signály nějak oddělit. K tomu se používá metoda CDMA (Code Division Multiple

Access), nazývaná také kódový multiplex. Pomocí CDMA je signál každé družice před vysíláním násoben pseudonáhodnou posloupností s hodnotami +1 nebo -1. Této posloupnosti říkáme PN kód (v anglicky psané literatuře Pseudo Random Noise Code). Doba trvání bitu kódu je přibližně 1 μs a je tedy výrazně menší než doba trvání datového bitu. To způsobí, že vysílaný signál je kódem tak rozbit, že vypadá jako šum.

Každá družice používá jiný kód, přičemž tyto kódy se vybírají z množiny Goldových posloupností. To jsou posloupnosti, které jsou vzájemně málo korelované a přitom vedlejší maxima autokorelační funkce jsou zanedbatelná.

Tyto vlastnosti jsou požadovány z těchto důvodů: Autokorelační funkce s malými vedlejšími maximy je výhodná na měření zpoždění signálu, což je nutný předpoklad dobrého fungování systému. Slabá vzájemná nekorelovanost je zase základem CDMA. V přijímači probíhá před vlastní demodulací BPSK nejprve přenásobení přijímaného signálu kódem té družice, jejíž signál chceme demodulovat. Přenásobení způsobí, že signál požadované družice se plně obnoví (protože 1 * 1 = 1 a − 1 * − 1 = 1 a tedy druhým násobením se úplně zruší vliv prvního násobení na družici). Signál nechtěné družice se díky nekorelovanosti neobnoví, a protože vypadá jako šum, je následnými obvody také jako šum potlačen.



Obrázek 3.2 – Formát dat GPS

3.3.2 Přesnost určení polohy

Míra přesnosti určení polohy podléhá několika vlivům, které lze stanovit. Protože chyba polohy je náhodná veličina, musí se její velikost popisovat nějakým jejím statistickým parametrem. V navigaci se obvykle používá efektivní hodnota chyby, což je odmocnina z průměru kvadrátu chyby.

Při určení polohy se nejprve změří vzdálenost přijímače od satelitů, dále se vypočtou polohy satelitů a nakonec je vypočítána poloha přijímače.

3.3.2.1 Přesnost měření vzdáleností

Největším faktorem ovlivňující přesnost měření vzdálenosti je atmosférický šum, který zkresluje signály a zabraňuje tak přesnému určení zpoždění. Jeho vliv způsobí na kmitočtu 1,57542 GHz (pro civilní sektor) chybu o směrodatné odchylce 7,5 m. Na kmitočtu 1,2276 GHz (pro vojenský sektor) je směrodatná odchylka této chyby 1,5 m.

Dalším faktorem je nepřesná znalost rychlosti šíření radiových vln, která je rovna rychlosti světla. Vlny se ale nešíří vakuem, ale atmosférou, v níž se obzvláště průchodem ionosférou rychlost mění. Tato změna však není konstantní, ale závisí na orientaci dráhy signálu a navíc se mění i stavem ionosféry v závislosti např. na ročním období, poloze Slunce apod. V přijímačích bývá zaimplementována funkce, která tyto změny zohledňuje. Směrodatná odchylka chyby vzdálenosti v důsledku tohoto faktoru je pro civilní sektor 5 - 10 m. Pro vojenský sektor je tato chyba výrazně menší, v důsledku toho, že může používat signálu na obou kmitočtech. Protože změna rychlosti ionosférou je frekvenčně závislá, lze ji z měření na dvou kmitočtech eliminovat.

Třetím faktorem je vícecestné šíření signálu. Pokud se přijímač pohybuje v zástavbě, přijímá signály jednak přímo od družice, ale také signály odražené. Velikost této chyby je závislá na terénu, v němž se přijímač nachází.

3.3.2.2 Přesnost určení polohy družic

Satelity vysílají parametry dráhy, na které kolem Země obíhají. Tyto parametry se nazývají efemeridy, což je astronomické přesné určení polohy kosmického tělesa v určitém čase, přesný údaj o čase, dále odhad zpoždění signálu v ionosféře a ještě celou řadu dalších údajů. Mimoto vysílají satelity tzv. almanac, což je vlastně databáze dalších satelitních stanic. Efemeridy jsou zjišťovány pozemními stanicemi, které sledují satelity a z jejich pohybu je předpovídají. Následně tyto parametry satelitům odesílají a ty je zařazují do svého vysílání.

Jsou proto možné dva druhy chyb – chyba v predikci efemeridu a chyba v pohybu satelitu způsobená např. nárazem meteoritu. Směrodatná odchylka chyby vzdálenosti v důsledku chyby polohy družice je přibližně 4 m a je přirozeně stejná pro civilní i vojenský sektor.

3.3.2.3 Vlastní výpočet polohy

Efektivní hodnota chyby určení polohy je dána součinem směrodatné odchylky určení vzdálenosti a koeficientu, který charakterizuje rozmístění družic na hemisféře (tento koeficient se nazývá DOP - dilution of precision, rozptyl přesnosti). Podobně efektivní hodnota horizontální chyby je dána součinem směrodatné odchylky určení vzdálenosti a koeficientu HDOP. Totéž platí pro efektivní hodnotu vertikální chyby, kde koeficient má označení VDOP.

Zatímco hodnota HDOP se mění se zeměpisnou polohou jen málo, hodnota VDOP se mění se zeměpisnou šířkou. V zeměpisné šířce ± 56° dosahuje svého minima a s dalším zvyšováním zeměpisné šířky pak výrazně roste. Tento nárůst chyby ve vyšších zeměpisných šířkách je způsoben tím, že po překročení zeměpisné šířky, která je rovna inklinaci dráhy, již satelity nedosahují nadhlavníku a kulminují ve stále nižších elevacích. Třídimenzionální chyba určení polohy prakticky sleduje průběh dominantní chyby výšky. V našich zeměpisných šířkách lze očekávat průměrné hodnoty DOP = 1,87, přičemž VDOP = 1,55 a HDOP = 1,05.

3.3.2.4 Shrnutí přesnosti

Sloučením všech vlivů způsobujících chybu určení vzdálenosti dostaneme směrodatnou odchylku vzdálenosti rovnou přibližně 12 metrům. Efektivní hodnota horizontální chyby v našich zeměpisných šířkách při 5° masce elevace a civilním uživateli je přibližně 12 metrů, zatímco efektivní vertikální chyba činí 19 metrů. To je však jen orientační číslo vycházející z průměrné hodnoty DOP.

Standardní GPS přijímač udává velikost chyby polohy. Tento údaj je bezpochyby přesnější, neboť přijímač zná aktuální DOP družic, jejichž signál používá k určení polohy. Přesto může nastat situace, kdy skutečná chyba může přesáhnout uváděnou hodnotu, protože skutečná chyba vzdálenosti převyšuje standardní hodnotu. Nejčastěji to bývá chyba vícecestného šíření, kterou přijímač nemůže odhalit, protože nedokáže určit, zda se uživatel nepohybuje v zástavbě, kde vznikají odrazy.

Výše uvedený postup určení polohy, tj. řešení soustavy rovnic s využitím měřením zpoždění v daný časový okamžik, je v přijímačích modifikován postupem, který bere v potaz i historii pohybu přijímače. Ten, jako každý fyzikální objekt, je ve svých pohybech limitován fyzikálními zákony, a proto lze ke stanovení polohy využít i měření z předchozích časových okamžiků a tím snížit chybu udávané polohy. Takovým postupem, který se v GPS přijímačích standardně používá, je Kalmanova funkce.

4 Analýza problému

4.1 Požadavky

4.1.1 Obecně – co by měl systém umožňovat

Systém by měl být schopen šířit informace v rámci uzavřené sítě. Tzn. pomocí PDA – klienta bychom měli být schopni získat od serveru multimediální a jiné informace na základě nějakého určujícího údaje, v našem případě aktuální pozicí na zemi – GPS pozici. Samotné šíření dat by mělo být zajištěno lokální bezdrátovou sítí (buď technologií Wi-fi nebo pomocí vysokorychlostního připojení mobilních operátorů), přes kterou by měla mít aplikace na kapesním počítači přístup k serveru, na kterém poběží naše serverová aplikace. Serverová aplikace by pak měla přijímat požadavky a náležitě je zpracovávat. Dále jsou rozebrány požadavky na jednotlivé části implementovaného systému.

4.1.2 PDA klient

Po spuštění klienta na kapesním počítači se zobrazí obrazovka s výpisem okolních objektů. Jakmile se přístroj přiblíží k nějaké zaznamenanému objektu, aplikace se přesune do popředí a uživateli oznámí informace o okolí – nalezené objekty. Jakmile si uživatel vybere nějaký objekt, který ho zajímá, aplikace zobrazí patřičné multimediální informace o tomto objektu. Tato data budou rozdělena na textová, audio a video. Mělo by být také možné po zobrazení seznamu dat zobrazit i výchozí textový článek, kde bude možné např. uvést základní bližší informace o vybraném objektu. O všech těchto operacích by měl být uživatel přehlednou formou informován.

4.1.3 Server

Serverová část systému by měla obsahovat program pro obsluhu požadavků od klienta a rovněž také administraci dat v systému. Měli bychom být schopni provádět následující operace:

Správa dat:

- Přidání objektu
- Odstranění objektu
- Změna hodnot objektu
- Přidání GPS souřadnice
- Odstranění GPS souřadnice
- Úprava GPS souřadnice
- Přidání textové objektu
- Odstranění textové objektu

- Změna hodnot textové objektu
- Přidání jednoduchých webových stránek
- Odstranění jednoduchých webových stránek
- Změna hodnot jednoduchých webových stránek
- Přidání audio souboru
- Odstranění audio souboru
- Změna hodnot audio souboru
- Přidání video souboru
- Odstranění video souboru
- Změna hodnot video souboru
- Přidání již existující textové informace
- Přidání již existujícího audio souboru
- Přidání již existujícího video souboru
- Přidání již existující jednoduché webové stránky

Správa GPS souřadnic na mobilním zařízení:

- Přidání GPS souřadnice na server
- Odstranění GPS souřadnice ze serveru
- Úprava GPS souřadnice na serveru
- Sejmutí GPS souřadnice z GPS zařízení

Administrační rozhraní by mělo být jednoduché a intuitivní, přičemž by je měl být schopen ovládat i člověk se základní znalostí práce na počítači. Uživatel – v tomto případě administrátor – by měl být rovněž přehledně informován o aktuálně probíhajících úlohách systému.

4.2 Analýza

Při analýze problému a možností systému byly v konečné fázi srovnávány dvě základní varianty, kterými lze hledaného cíle dosáhnout. Asi tím nejjednodušším způsobem by bylo zřízení webových stránek na serverovém počítači a na PDA povolení přístupu k těmto stránkám (pokud by se jednalo o připojení Wi-fi). V té chvíli ale mizí kontrola nad tím, co se s multimediálními soubory po stažení děje, navíc je uživatel dotazován, co se souborem má internetový prohlížeč provést apod. Tímto může být uživatel zatížen a mohlo by se např. stát, že nebude vědět, jakou volbu má potvrdit. Další možností, která se jeví při dosavadních možnostech jako nejvhodnější, je vytvoření vlastní aplikace, která tuto činnost bude vykonávat za uživatele, tzn. že dojde ke zjednodušení a zefektivnění celého systému ve vztahu k uživateli a systém se tak stane „user friendly“.

Vzhledem k uvedeným skutečnostem byla pro řešení zadaného problému implementována varianta vytvoření vlastní aplikace. Jak již bylo zmíněno dříve, právě přechod od implementace webových stránek HTML k vývoji aplikace, která využívá všech vymožeností a schopností grafického rozhraní operačního systému, je v současné době novým trendem tvorby business aplikací. Uživatel totiž nemusí ovládat internetový prohlížeč, ale plnohodnotnou aplikaci, která mu poskytne totéž, na co je zvyklý ze svého počítače doma.

Co se týče uložení dat na serveru, jeví se nejvhodnějším řešení s využitím SQL serveru pro databázové informace a IIS (Internetové Informační Služby) systému Microsoft Windows pro uložení souborů a obslužného serveru pro klienta PDA. V případě výběru SQL serveru, po zvážení nutných a dostačujících podmínek, a to i s výhledem na další možné rozšíření systému by ke splnění zadaného úkolu měl plně stačit Microsoft SQL Server 2005 ve verzi Express Edition. Tato verze Express Edition je navíc zcela zdarma, přičemž se jedná o velice výkonnou databázovou aplikaci ochuzenou pouze o některé funkce na nejvyšší úrovni, které naše aplikace stejně nevyužije.

4.2.1 UML

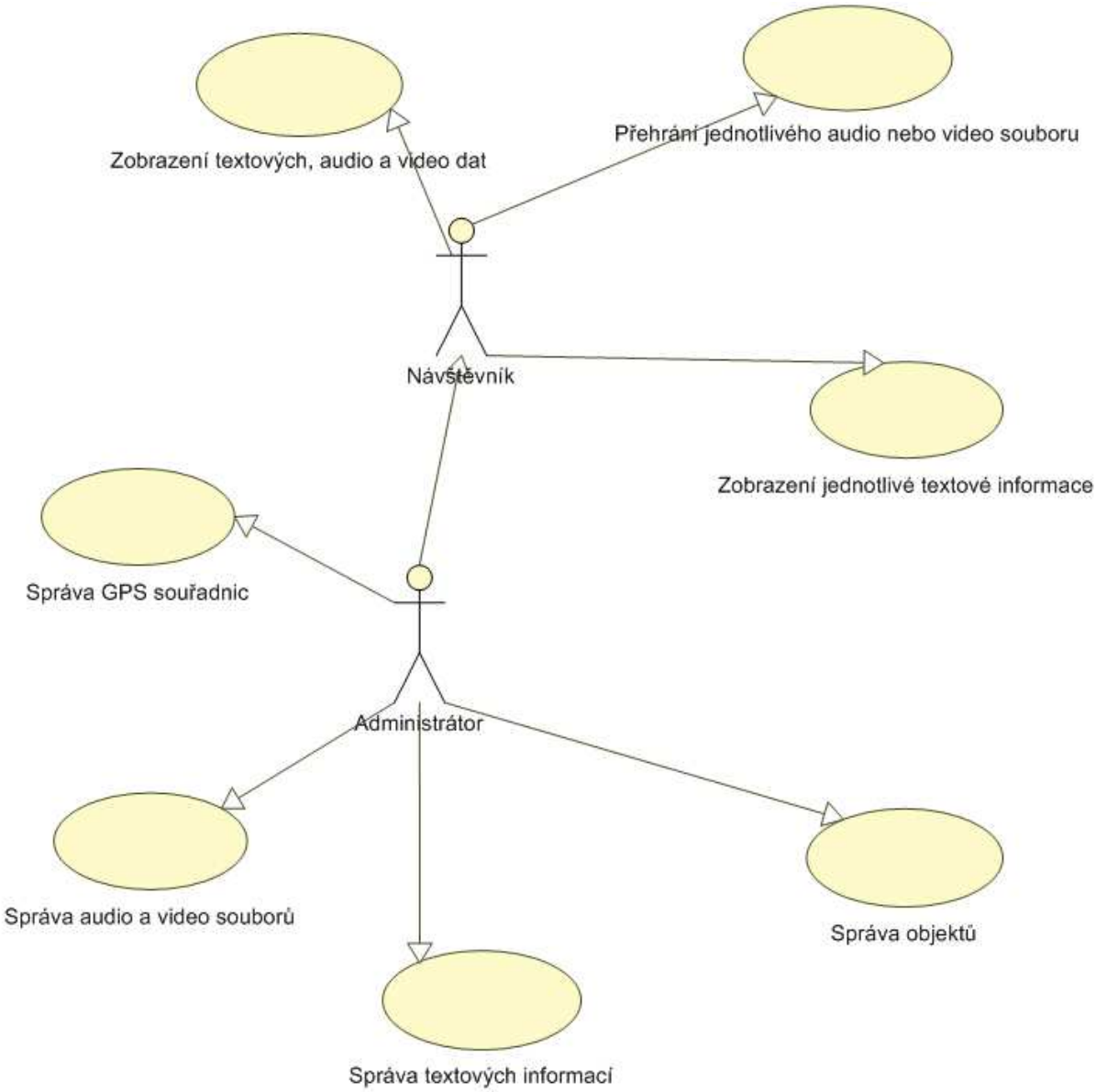
4.2.1.1 Co je UML

Unified Modeling Language (viz [15]) je v softwarovém inženýrství grafický jazyk pro vizualizaci, specifikaci, navrhování a dokumentaci programových systémů. UML nabízí standardní způsob zápisu, a to jak návrhů systému včetně konceptuálních prvků jako jsou business procesy a systémové funkce, tak i konkrétních prvků jako jsou příkazy programovacího jazyka, databázová schémata a znovupoužitelné programové komponenty.

UML podporuje objektově orientovaný přístup k analýze, návrhu a popisu programových systémů. UML neobsahuje způsob, jak se má používat, ani neobsahuje metodiku(y), jak analyzovat, specifikovat či navrhovat programové systémy. Standard UML definuje standardizační skupina Object Management Group (OMG).

4.2.1.2 Use Case – model jednání

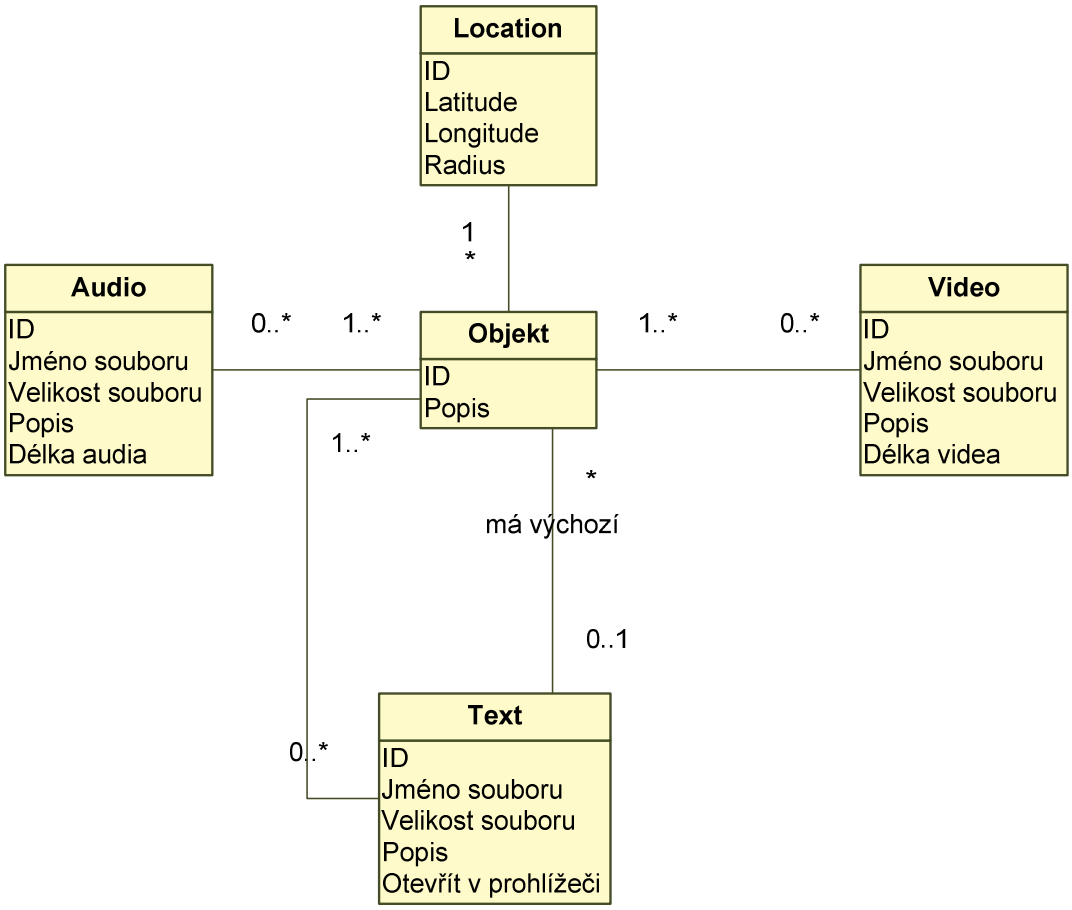
V modelu jednání jsou uvedeni všichni uživatelé, kteří do systému vstupují. V našem případě je to tedy návštěvník města a administrátor systému, přičemž administrátor má tytéž práva jako návštěvník (což je znázorněno vztahem generalizace/specializace) a navíc ještě práva pro správu dat v systému.



Obrázek 4.1 – Use Case model

ER-diagram – diagram tříd

V následujícím diagramu jsou zobrazeny vztahy mezi jednotlivými třídami (objekty).

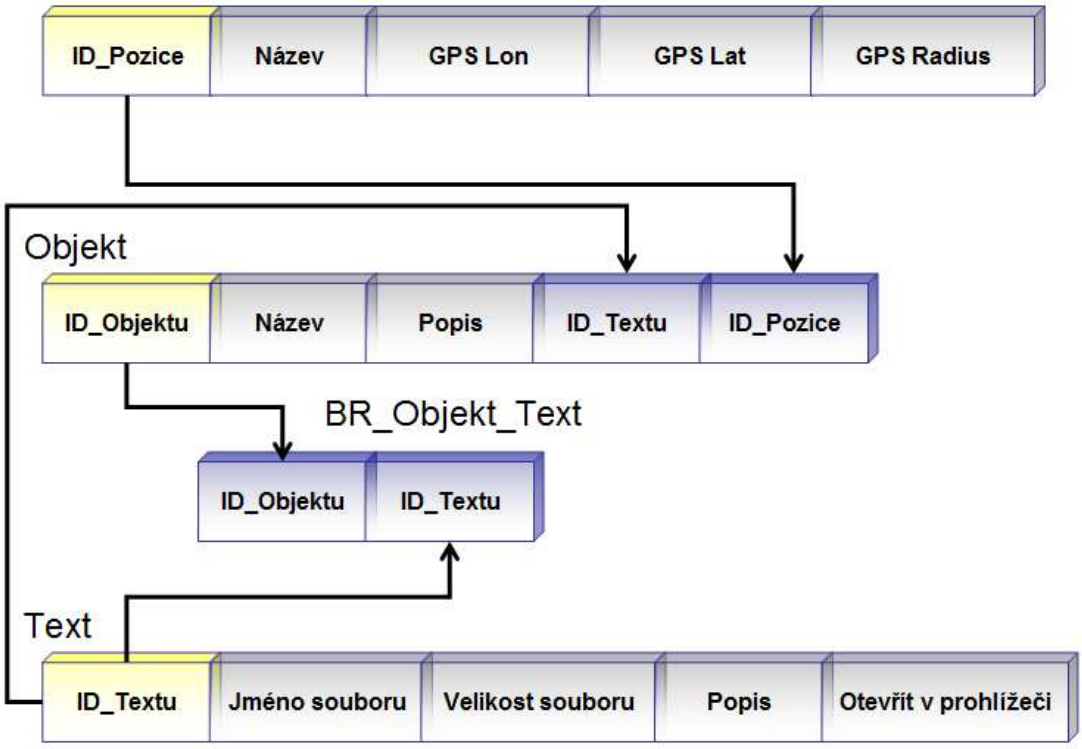


Obrázek 4.2– Diagram tříd

4.2.1.3 Ukázka převodu modelu na tabulku v databázi

Při převodu modelu na tabulky v databázi se využívá proces normalizace, tedy zjednodušování a přizpůsobování databázových struktur do podoby výhodné pro implementaci v relační databázi.

Pro ukázku byl vybrán vztah Objekt – Text, přičemž 1 až x objektů může obsahovat 0 až x textů a každý objekt má 0 až 1 výchozích textů. Jeden objekt také obsahuje 0 až 1 GPS souřadnici – Location.



Obrázek 4.3 – Ukázka převodu modelu na tabulku v databázi – text

4.2.1.4 Location

Uložená pozice na Zemi v podobě souřadnice GPS – délku a šířku s radiusem (jak široké okolí má být bráno jako součást souřadnice).

4.2.1.5 Objekt

Informace o všech objektech v systému jsou uchovávány v tabulce Objekt – Item. Objekt zde má uvedeny popis a výchozí textovou položku přes cizí klíč ID_Textu. Rovněž je zde návaznost na umístění objektu na Zemi přes cizí klíč ID_Pozice. Popisem je myšlen název objektu, případně další informace, které uživatel uzná za vhodné. Tento popis je de facto pouze pro administrační část systému, na klientovi se tento údaj nezobrazuje.

4.2.1.6 Text

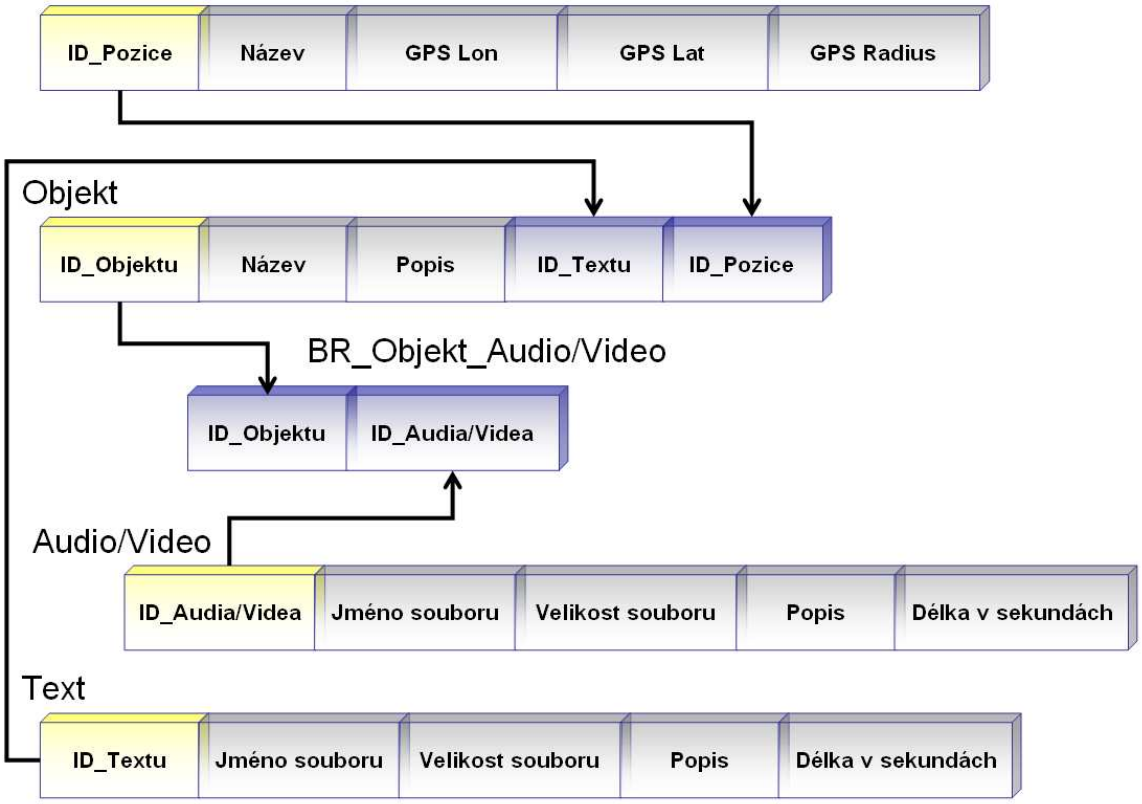
Textové položky jsou uloženy v tabulce text, přičemž se vede záznam o popisu této položky, jménu souboru, jeho velikosti v bytech a o tom, zda se má otevřít rovnou v prohlížeči, nebo má být stažen na disk a až potom zobrazen.

4.2.1.7 Audio

Audio položky jsou uloženy v tabulce audio a uchovávají se informace o popisu audio souboru, jeho jméně, velikosti v bytech a délce v sekundách.

4.2.1.8 Video

Video soubory se uchovávají v tabulce video a struktura informací o nich je stejná jako u audio položek.



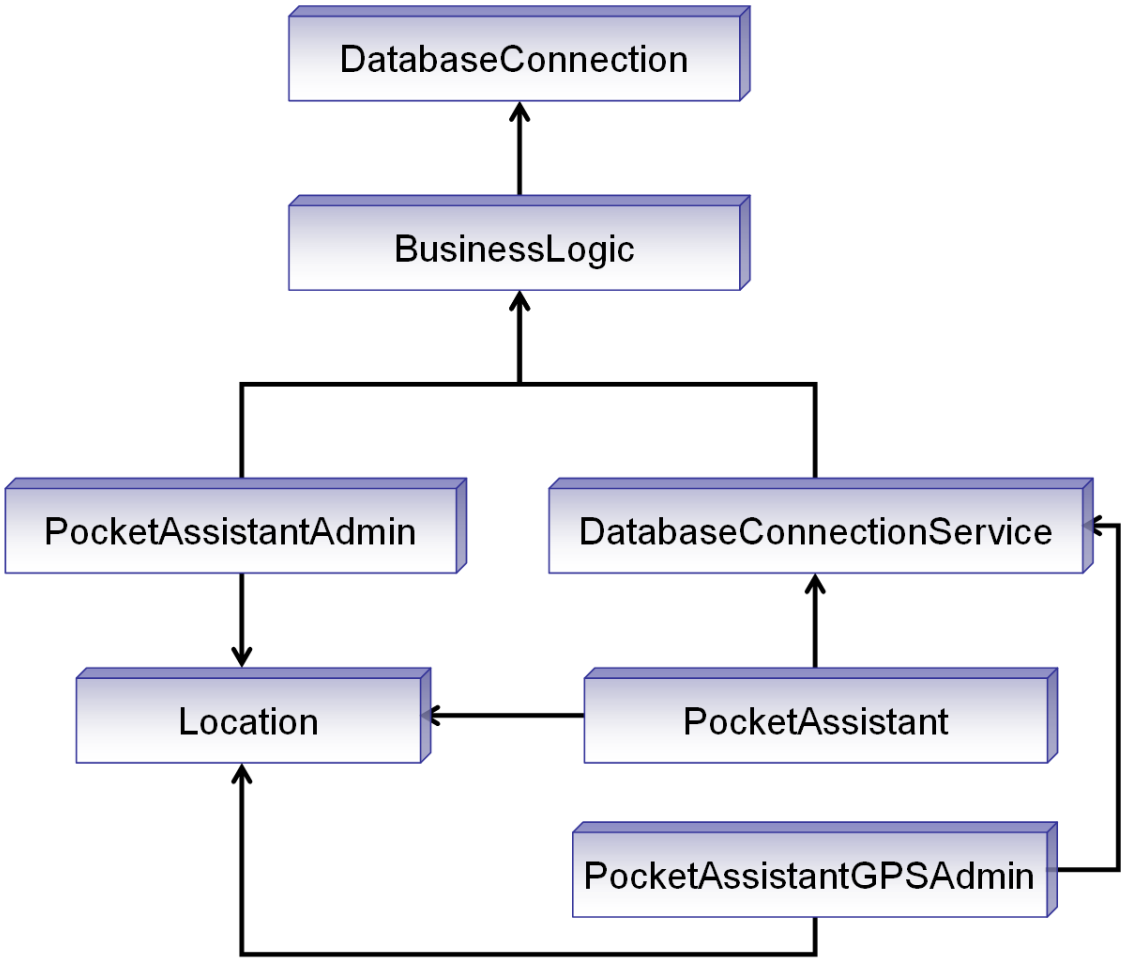
Obrázek 4.4 – Ukázka převodu modelu na tabulku v databázi – audio/video

5 Implementace

Jak je již zmíněno v zadání, systém je implementován v programovacím jazyce C#, a to jak klient na PDA, tak serverová část na PC. K vývoji byla použita nejnovější verze zmíněného programovacího jazyka C# ve verzi 3.5. Oproti starší verzi 1.1 a 2.0 zde došlo především k výrazným pokrokům v technologiích a komponentách, které jsou pro aplikaci nutné. Rovněž samotné vývojové prostředí bylo značně vylepšeno a poskytuje nyní více prostředků, které poměrně značně zjednodušují a zefektivňují práci programátora. V následujících podkapitolách budou probrány jednotlivé implementované části systému, tedy klient a server.

5.1 Popis řešení jednotlivých projektů

Systém se dělí na klientskou část a na část serverovou. Na následujícím obrázku je celá architektura projektu, která bude nyní ve stručnosti popsána.



Obrázek 5.1 – Architektura projektu

5.1.1 DatabaseConnection

Projekt je navržen jako knihovna dll, která zajišťuje přístup do databáze. Samotný přístup do databáze definuje interface DatabaseDriver, který je pak implementován pro databáze Microsoft Access, Microsoft SQL a Oracle. Tímto je dosažena možnost připojení aplikací k různým druhům databázi. Jako výchozí je dále používána databáze Microsoft SQL ve verzi Express.

5.1.2 BusinessLogic

Projekt je navržen jako knihovna dll, která zprostředkovává práci s databází. Přístup do databáze je prováděn přes předchozí DatabaseConnection.

Knihovna obsahuje práci se všemi prvky databáze – tedy s objekty, texty, audio soubory, video soubory, GPS souřadnicemi. Pro práci s databází tedy při využití této knihovny není potřeba používat pokaždé připojení k databázi apod., knihovna toto vše řeší za uživatele. Konfigurace k připojení do databáze je uložena v souboru XML jméno_exe_souboru_aplikace_která_knihovnu využívá.exe.config.

Jednotlivá nastavení:

Jméno nastavení	**Popis**
SERVER	typ připojení např. mssql – Microsoft SQL
SQLSERVER_USER	přihlašovací jméno uživatele do databáze (např. sa)
SQLSERVER_PASSWORD	přihlašovací heslo uživatele do databáze (např. sa)
SQLSERVER_SERVER	adresa server (např. localhost/SQLEXPRESS)
SQLSERVER_DB	jméno databáze (např. PocketAssistant)
TEXT_FOLDER	cesta k adresáři, kde budou uchovávány textové soubory (obvykle umístěna v adresáři s webovou službou (viz dále)
AUDIO_FOLDER	cesta k adresáři, kde budou uchovávány audio soubory (obvykle umístěna v adresáři s webovou službou (viz dále)
VIDEO_FOLDER	cesta k adresáři, kde budou uchovávány video soubory (obvykle umístěna v adresáři s webovou službou (viz dále)

5.1.3 DatabaseConnectionService

Projekt zajišťující provoz webové služby a komunikaci s databází. Obsahuje procedury pro práci s databází přes projekt BusinessLogic. Zároveň generuje KML soubor s definicí objektů, které jsou pak zobrazitelné na mapách společnosti Google. Bližší informace ke KML souborům a jejich implementaci jsou rozvedeny v kapitole 5.3.2 Google maps – generování KML souboru.

5.1.4 Location

Projekt zajišťuje práci s GPS zařízením. Umožňuje uchytit události na změnu polohy, změnu stavu přístroje apod. Zapouzdřuje tak celou práci s GPS do jedné knihovny, kterou pak již stačí jen přilinkovat a pracovat s GPS zařízením jednoduše. Komunikace probíhá přes Microsoft Intermediate Driver.

Obsahuje i datovou strukturu GPS pozice uložené v databázi, čímž umožňuje pozdější cachování pozic v aplikaci. Poskytuje rovněž procedury pro převod mezi souřadnými jednotkami a zjišťování vzdálenosti dvou bodů apod.

5.1.5 PocketAssistant

Hlavní aplikace projektu – klientská část běžící na zařízení Pocket PC, umožňující získávat multimediální informace o okolních objektech. Využívá všechny předchozí projekty. Dělí se základně na dvě obrazovky – obrazovku s výpisem okolních objektů a detail objektu. Pro audio a video soubory spouští navíc Windows Media Player. Detaily obrazovek a bližší popis je k dispozici v kapitole 5.2.5 – Popis implementace uživatelského rozhraní.

5.1.6 PocketAssistantAdmin

Aplikace pro správu dat – serverová část běžící přímo na serveru. K serveru se připojuje přímo pomocí BusinessLogic. Pokud jsou přístroje připojeny přes uzavřenou síť, pak může být server umístěný v kterékoliv části sítě. Nicméně Google Maps vyžadují umístění na veřejné síti, což se dá obejít vygenerováním KML souboru a jeho statickým umístěním na kterémkoliv hostingu. Bližší informace o projektu a ukázky jednotlivých obrazovek jsou k vidění v kapitole 5.3.3 – Popis implementace uživatelského rozhraní

5.1.7 PocketAssistantGPSAdmin

Aplikace pro správu GPS pozic přímo na Pocket PC zařízení. Využívá připojení k serveru webové služby – tzn. projekt ConnectionService a dále využívá projekt Location pro práci s GPS zařízením.

5.1.8 Spolupráce s Google maps pro mobilní aplikace

Společnost Google nabízí volně ke stažení aplikaci Google maps, která slouží k zobrazení map na všech různých zařízeních – od osobního počítače až po Pocket PC / mobilní telefony. V nejnovější verzi podporuje i zobrazení aktuální GPS pozice na mapě.

Ve spojení s KML soubory tak umožňuje přehledně zobrazit všechny objekty na mapě se základními informacemi o nich.

V následující kapitole jsou probrány jednotlivé části architektury klient–server podrobněji.

5.2 Klient

Klient je implementován za pomoci .NET rozhraní pro mobilní aplikace, tzv. .NET Compact Framework. Jako součást Visual Studia 2005 i v nejnovější verzi 2008, je k dispozici simulační nástroj kapesního přístroje PDA. Právě v novější verzi 2008 došlo k jeho vylepšení, a to jak v uživatelském rozhraní, tak i ve verzi simulovaného operačního systému (již je nabízena i verze 6.0).

Architektura klient-server je implementována za pomocí technologie webové služby, kdy klient s její pomocí komunikuje s SQL serverem a zobrazuje patřičné informace uživateli. Obě použité technologie budou detailněji rozebrány dále v sekci implementace serverové části.

Klient je implementován již v předchozích kapitolách probranou technologií tzv. tenkého klienta, kdy zobrazuje pouze výsledky operací vykonaných na serverové části, případně tyto výsledky analyzuje a podle nich pak zvolí dané chování.

V následujících podkapitolách probereme způsob implementace klienta včetně uživatelského rozhraní a instalace.

5.2.1 Spolupráce s GPS zařízením

GPS zařízení komunikuje s přístrojem obvykle pomoci některého COM portu (tzn. sériovým rozhraním). GPS vysílá signály, které informují o dané poloze a dalších určujících údajích. Ve Visual Studiu 2008 a emulátoru Windows Mobile verze 6.0 je možné nainstalovat aplikaci, která simuluje připojené GPS zařízení. Je však nutné mu předložit NMEA log soubor, který obsahuje zaznamenanou komunikaci mezi GPS zařízením a jeho konzumentem.

Samotná komunikace pak probíhá přes GPS Intermediate Driver, který vytváří vrstvu mezi GPS zařízením a operačním systémem Windows Mobile, což umožňuje spolupracovat s GPS v aplikaci bez rozdílu mezi výrobci GPS zařízení. GPS Intermediate Driver využívá nastavení v registru, které udává, na kterém COM portu je GPS zařízení připojeno. Toto nastavení je možno upravit pomocí aplikace GPS Settings (Nastavení GPS).

5.2.1.1 NMEA data

NMEA je zkratkou National Marine Electronic Association, což je společnost, která definovala rozhraní mezi různými elektronickými zařízeními. Tento standard tak umožňuje zasílat různé informace do počítače nebo dalším elektronickým zařízením, které na tomto standardu fungují.

Rozhraní je navrženo tak, aby se jím dala posílat data ve sledu po sobě, ale vzájemně nezávisle. Pro každou kategorii je možné definovat různé bloky dat, které mohou odlišovat např. různé výrobce zařízení. První dva znaky bloku dat definují zařízení, které bloky vysílá (pro GPS jsou to znaky GP).

Následující tři znaky definují druh obsahu bloku dat. NMEA navíc umožňuje výrobcům definovat vlastní bloky dat za účelem, který vyžadují. Všechny tyto vlastní bloky dat začínají písmenem P následované třemi písmeny identifikující výrobce zařízení. Např. GPS přijímač Garmin používá označení bloku dat PGRM, Magellan používá PMGN.

Každý blok dat začíná znakem $ a končí koncem řádky (CRLF). Blok dat může být dlouhý maximálně 80 znaků viditelného textu. Data jsou v jednom řádku oddělena čárkami. Samotná data jsou pouze ASCII znaky. Na konec je zpravidla pak přidán kontrolní součet – za znak * se přidají 2 znaky, které představují hexadecimální kontrolní součet exklusivní operací OR všech znaků mezi znaky $ a *. Tento kontrolní součet je u některých bloků dat vyžadován.

V průběhu vývoje verze 1.0 došla přes verze 1.5, 2.0, 2.3 až do verze 3.01, nicméně u GPS zařízení se používá momentálně převážně verze 2.0.

5.2.1.2 Hardware

Komunikace probíhá pomocí RS232 protokolu (nicméně pokud by se měl striktně dodržet standard, komunikace by měla probíhat prostřednictvím EIA-422 protokolu). Rychlost přenosu dat by měla být nastavena na hodnotu 4800 b/s s osmibitovými daty bez parity a s jedním stop bitem. Tato rychlost by měla byt dostačující k tomu, aby bylo možné vysílat informace o poloze každou sekundu, nicméně některá zařízení tuto rychlost nesplňují, a tak dokážou zpracovat data např. každou druhou sekudnu – tím dochází někdy k občasnému zpoždění.

Při rychlosti 4800 b/s je možné odeslat pouze 480 znaků za sekundu, přičemž NMEA formát bloku dat má délku 82 znaků, čímž je možné do jedné zprávy vměstnat 6 různých bloků dat.

5.2.1.3 Zprávy zasílané GPS zařízením ve formátu NMEA 2.0

První dva znaky, jak již bylo popsáno, definují zařízení – v našem případě GPS – tedy značka je GP. Další tři znaky definují obsah, který bude detailněji u nejčastějších zpráv probrán v následující tabulce.

Značka	Popis
GPAPB	Auto Pilot B
GPBOD	směr, od počátku do cíle
GPGGA	oprava dat
GPGLL	Lat/Lon data
GPGSA	celková informace o přijímaných datech ze satelitu
GPGSV	detailní informace o satelitu
GPRMB	minimum doporučených dat při sledování cesty
GPRMC	minimum doporučených dat
GPRTE	informace o cestě, jen pokud je nějaká aktivní
GPWPL	data o cílovém bodě, jen pokud je nějaká cesta aktivní

V následujících odstavcích si rozebereme detailně jednotlivé značky.

GGA

Oprava dat, která poskytuje 3D polohu a upřesňuje již získaná data.

```
$GPGGA,123519,4807.038,N,01131.000,E,1,08,0.9,545.4,M,46.9,M,,*47

Kde:
     GGA          určení polohy globálního pozičního systému
     123519       odpovídající čas 12:35:19 UTC
     4807.038,N   zeměpisná šířka 48 stupňů 07.038' N
     01131.000,E  zeměpisná délka 11 stupňů 31.000' E
     1            přesnost dat: 0 = neplatná data
                                1 = GPS určení polohy (SPS)
                                2 = DGPS určení polohy
                                3 = PPS určení polohy
                                4 = Real Time Kinematic
                                5 = Float RTK
                                6 = odhad (mrtvý výpočet)
                                7 = manuální vstupní mód
                                8 = simulační mód
     08           počet sledovaných satelitů
     0.9          horizontální snížení přesnosti (HDOP) v metrech
     545.4,M      výška nad elipsoidem
     46.9,M       výška geoidu nad WGS84 elipsoidem
        (prázdné pole)  stáří poslední aktualizace DGPS v sekundách
        (prázdné pole)  ID stanice DGPS
        *47             kontrolní součet, vždy začíná znakem *
```

GSV

Satelity ve výhledu podávají informace o viditelných satelitech a umožňují tato data sledovat. Dokáží ale informovat v jedné zprávě maximálně o 4 satelitech.

```
$GPGSV,2,1,08,01,40,083,46,02,17,308,41,12,07,344,39,14,22,228,45*75
```

```
Kde:
     GSV          satelity v dohledu
     2            počet vět nutný k získání kompletní informace
     1            věta 1 z celkového počtu 2
     08           počet satelitů v dohledu

     01           identifikační číslo satelitu
     40           elevace ve stupních
     083          azimut ve stupních
     46           SNR (odstup signálu od šumu) – vyšší hodnota je lepší
     *75          kontrolní součet
```

RMC

Informace o poloze, rychlosti pohybu a čase.

```
$GPRMC,123519,A,4807.038,N,01131.000,E,022.4,084.4,230394,003.1,W*6A
```

```
Kde:
     RMC          Recommended Minimum (doporučené minimum)
     123519       odpovídající čas 12:35:19 UTC
     A            stav A = aktivní nebo V = neplatný
     4807.038,N   zeměpisná šířka 48 stupňů 07.038' N
     01131.000,E  zeměpisná délka 11 stupňů 31.000' E
     022.4        rychlost v uzlech
     084.4        kurz pohybu ve stupních
     230394       datum – 23. březen 1994
     003.1,W      magnetická deklinace ve stupních
     *6A          kontrolní součet
```

GLL

Informace o poloze a čase.

```
$GPGLL,4916.45,N,12311.12,W,225444,A,*31
```

```
Kde:
     GLL          geografická pozice, zeměpisná šířka a délka
     4916.46,N    zeměpisná šířka 49 stupňů 16.45 minut N
     12311.12,W   zeměpisná délka 123 stupňů 11.12 minut W
     225444       odpovídající čas 22:54:44 UTC
     A            příznak Aktivních nebo Void (neplatných) dat
     *31          kontrolní součet
```

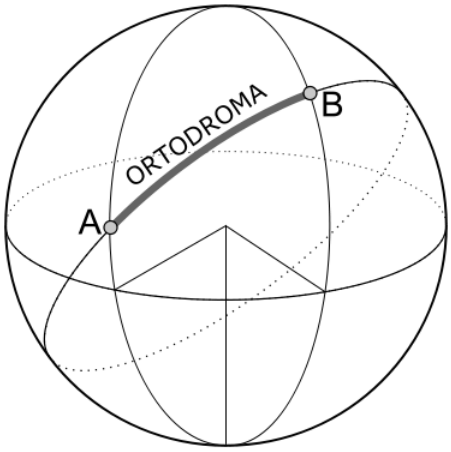
Tabulka a popisy viz [2].

5.2.2 Výpočet vzdálenosti dvou bodů na Zemi

Pro výpočet vzdálenosti dvou bodů na Zemi se pro zjednodušení pracuje s body na povrchu referenční koule. V praxi se využívají nejvíce dvě metody – Ortodroma a Loxodroma. Obě tyto metody budou podrobněji popsány v následujících podkapitolách.

5.2.2.1 Ortodroma

Z řeckého ortos – přímý, dromos – cesta. Grafické znázornění viz. Obrázek 5.2– Ortodroma.



Obrázek 5.2– Ortodroma

Využívá se k výpočtu nejkratší spojnice dvou bodů na referenční kulové ploše. Tvoří ji kratší oblouk kružnice se středem ve středu Země. Ortodroma je sice sice nejkratší spojnicí dvou bodů, v navigaci je ale přesnější loxodroma. Její dráha totiž udržuje stále stejný úhel s poledníkem (azimut), na rozdíl od ortodromy, u které se azimut obecně mění. Nicméně pro vzdálenosti do 800-1000 km je rozdíl mezi výpočtem ortodromy a loxodromy zanedbatelný, proto si pro naše účely vystačíme s výpočtem s pomocí ortodromy.

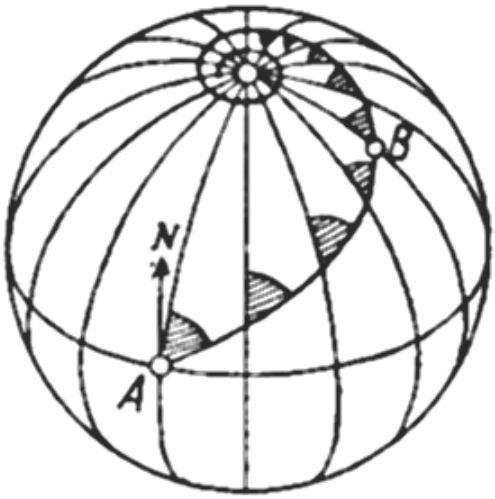
Výpočet délky ortodromy vychází ze sférické trigonometrie. Označme $[\varphi_1;\lambda_1]$ a $[\varphi_2;\lambda_2]$ souřadnice krajních bodů ortodromy a σ její délku. R značí poloměr referenční koule – v našem případě to bude poloměr Země. V jednotkách (metrech / kilometrech), ve kterých poloměr dosadíme do rovnice, bude i výsledná hodnota. Délku pak můžeme ze sférické kosinové věty definovat dle Rovnice 1 – výpočet vzdálenosti pomocí ortodoxy:

$$\sigma = R.\arccos\left(\sin\varphi_1 \sin\varphi_2 + \cos\varphi_1 \cos\varphi_2 \cos\left(\lambda_2 - \lambda_1\right)\right)$$

Rovnice 1 – výpočet vzdálenosti pomocí ortodoxy

5.2.2.2 Loxodroma

Z řeckého lox/loxo – šikmý, dromos – cesta. Grafické znázornění viz. Obrázek 5.3 – Loxodroma.



Obrázek 5.3 – Loxodroma

Křivka na referenční kulové ploše, která protíná všechny poledníky pod stejným úhlem. Přestože loxodroma není nejkratší spojnicí dvou míst na referenční ploše, byly loxodromické cesty v minulosti využívány při námořní plavbě. Pro svou jednoduchost jsou loxodromické cesty používány i dnes v námořní a letecké navigaci.

Délka se pak vypočte podle dvou vzorců v závislosti na azimutu loxodromy A. Pro A nerovno 90° je výpočet pro uveden dle rovnice Rovnice 2 – výpočet vzdálenosti pomocí loxodromy pro A != 90°.

$$\sigma = \frac{R}{\cos A}(\varphi_2 - \varphi_1)$$

Rovnice 2 – výpočet vzdálenosti pomocí loxodromy pro A != 90°

Pro A rovno 90° je výpočet uveden dle rovnice Rovnice 3 – výpočet vzdálenosti pomocí loxodromy pro A = 90°.

$$\sigma = R.\cos U(\lambda_2 - \lambda_1)$$

Rovnice 3 – výpočet vzdálenosti pomocí loxodromy pro A = 90°

5.2.2.3 Vztah mezi ortodromou a loxodromou

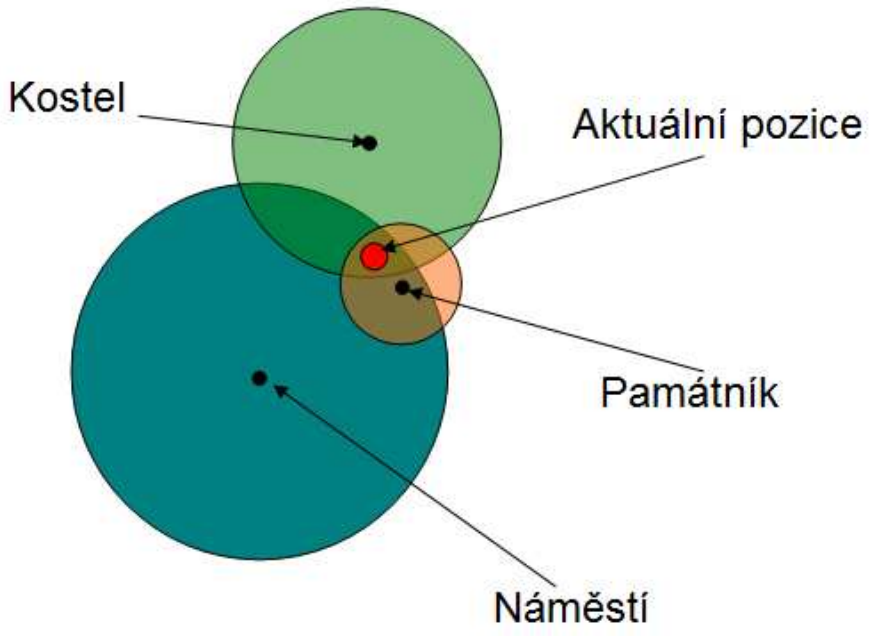
- délka loxodromy mezi dvěma body je vždy větší nebo rovna délce ortodromy
- loxodroma a ortodroma jsou stejně dlouhé, pokud oba zvolené body leží na rovníku nebo pokud je azimut roven velikosti 0° či 180° (tedy loxodroma odpovídá poledníku)
- největší rozdíl mezi délkami ortodromy a loxodromy nastává ve chvíli, kdy zvolené body leží na stejné rovnoběžce (kromě rovníku) a azimut je tedy roven 90° nebo 270°
- na severní polokouli je loxodroma jižněji než ortodroma, na jižní polokouli je tomu naopak

5.2.3 Popis implementace

Aplikace po spuštění naváže spojení se vzdálenou webovou službou a do cache si stáhne všechny GPS pozice uvedené v systému, aby při každé změně polohy nemusela kontaktovat vzdálený server. Cache GPS pozic se pak ve výchozím nastavení obnovuje pravidelně po patnácti minutách.

Po stažení GPS pozic do cache aplikace následně naváže spojení s GPS přijímačem přes GPS Intermediate Driver a poté v případě změny polohy zkontroluje, zda se v okolí nenachází nějaký objekt. Pokud ano, tak naváže spojení se serverem a získá seznam objektů v okolí, který pak nabídne uživateli k výběru (detaily obrazovek jsou vyobrazeny v dále uvedené kapitole). Aplikace udržuje seznam již zobrazených objektů v rámci dané oblasti a pokud najde nějaký nový objekt v okolí, upozorní na tuto událost uživatele zvukovou výstrahou, případně i vibracemi podporuje-li zařízení tuto možnost a nový objekt se přenese do popředí. Uživateli tedy nabídne všechny objekty v okolí a pozastaví vyhledávání dalších objektů.

Jelikož jsou GPS souřadnice uloženy i s definicí velikostí okolí, je tak možné nadefinovat např. náměstí o velikosti 90 metrů v okolí, a tak není nutné dojít přesně do bodu určení. Zároveň je tak možné při průchodu mezi body, překrytím jejich okolí s aktuální pozicí, získat seznam okolních objektů a tím tak uživateli snadněji přiblížit okolní situaci.



Obrázek 5.4 – Ukázka určení okolí

Na předchozím obrázku, Obrázek 5.4 – Ukázka určení okolí je vidět jak okolí bodu funguje. Na aktuální pozici se jako okolí zobrazí památník, kostel a náměstí, a to v uvedeném pořadí (podle vzdálenosti – od nejbližšího k nejvzdálenějšímu). Uživatel tak má možnost zjistit informace o svém celém aktuálním okolí, tedy nejen informace o nejbližším okolí – památníku. Tímto způsobem je například možné při prohlídce Brna nadefinovat Brno jako velkou plochu, tzn. zvolit oblast podle rozlohy, a informace o Brně pak budou dostupné při pohybu ve vymezené oblasti v seznamu okolních objektů neustále.

5.2.3.1 Měření vzdálenosti – získání objektů v okolí

Při prohledávání okolí se využívá znalosti, které jsou uvedeny v kapitole 5.2.2 Výpočet vzdálenosti dvou bodů na Zemi. K výpočtu byla zvolena metoda ortodromy.

Pomocí dotazovacího jazyka LINQ vypadá dotaz na nalezení okolních GPS souřadnic zaznamenaných v systému následovně:

```
GPSPositions.Select(n =>
(6378137 * Math.Acos((Math.Sin(Math.PI * latitude / 180) *
Math.Sin(Math.PI * n.Latitude / 180)) +
(Math.Cos(Math.PI * latitude / 180) * Math.Cos(Math.PI * n.Latitude / 180)
* Math.Cos(Math.PI * n.Longitude / 180 - Math.PI * longitude / 180))) <=
n.Radius))
```

Kde,
latitude = latitude složka aktuální pozice
longitude = longitude složka aktuální pozice

Tyto dvě složky jsou násobeny konstantou Pi a děleny 180 stupni za účelem převodu do jednotek radiánu.

Proměnná n značí hledaný vybraný prvek ze seznamu prvků GPSPositions. Zbylá část dotazu se, jak je vidět, podobá dotazu v jazyce SQL, nicméně tento dotaz je aplikovatelný nejen na SQL databázi, ale i na seznam objektů, XML soubor apod.

Jako poslední je zadána podmínka vzdálenosti menší než u daného objektu (<= n.Radius). LINQ po provedení tohoto dotaz vrátí jako výsledek seznam objektů, které vyhovují uvedené podmínce.

5.2.4 Popis implementace vláken

Při spuštění aplikace se vytvoří obvykle jedno vlákno, které se stará o veškerý běh aplikace. Pokud aplikace provádí nějakou časově náročnější operaci, jako například navazování spojení se serverem a stahování multimediálních informací o okolí, tak se celá aplikace dostane do stavu, kdy nereaguje na žádné vstupy od uživatele, což obvykle nepůsobí moc dobře.

Proto je při spouštění takových časově náročných operací vhodné použít neblokující konstrukce. Tzn. operace je pak spuštěna na pozadí a grafické uživatelské rozhraní není po dobu operace blokováno. Tato vlastnost pak přináší mnoho výhod pro uživatele, který tak není obtěžován pozastavením aplikace a neschopností její činnosti v průběhu vykonávání operace. Ztěžuje ovšem práci pro programátora aplikace, protože je v té chvíli nutné tato vlákna mezi sebou synchronizovat a zabraňovat vstupu vláken do stejných částí programu atp. U aplikací s formuláři je navíc problém přístupu do formuláře a změny uživatelského rozhraní, které běží v jiném vláknu, než ve kterém se provádí časově náročná operace a odkud je tedy nutné informovat uživatele o jejím průběhu. Tento problém je v desktopových aplikacích vyřešen pomocí třídy BackgroundWorker, jak si ukážeme dále.

V jazyku C# existuje několik možností jak zajistit běh nějaké funkce na pozadí. První asi nejjednodušší možnost je využít konstrukce Timeru (časovače), který spouští operace uvedené v těle obslužné funkce v zadaných časových intervalech. Tato možnost se ale pro naše účely příliš nehodí, její použití je vhodnější na opakované provádění operací, což není náš případ.

Další možností je využít třídy Thread, která v sobě zapouzdřuje základní obsluhu a vyvolání vláken. Zde se již pro každé spuštění funkce vytvoří jedna instance objektu a ta je pak asynchronně zavolána a vykonána. Tato konstrukce ale neposkytuje žádnou větší kontrolu nad více vlákny a proto je vhodné využít některou z následujících konstrukcí.

Pro více vláken existuje v jazyku C# třída ThreadPool, která již zabezpečuje práci s více vlákny a je proto pro naše účely nejvhodnějším řešením. Její nástavbou je třída BackgroundWorker, která využívá jejích vlastností a řeší některé problémy, které souvisí s použitím vláken v aplikacích, které si probereme dále. Třída BackgroundWorker bohužel v .NET Compact frameworku není a proto ji bylo nutné v nějaké obdobné formě implementovat.

5.2.4.1 Třída Thread

Třída umožňuje pouze základní práci s vlákny. Umožňuje vlákno vytvořit a kontrolovat jeho průběh pomocí funkcí pro uspání a znovuzavedení vlákna a samozřejmě umožňuje vlákno explicitně ukončit.

Každému takto spuštěnému vláknu je navíc možné přidat prioritu, jejíž výchozí hodnota je nastavena na normální. Při komunikaci s rozhraním aplikace je nutné rozhraní volat přes metodu Invoke, jinak aplikace ohlásí chybu o neoprávněném přístupu.

5.2.4.2 Třída ThreadPool

Třída zabezpečuje práci s více vlákny, tzn. umožňuje vlákna řadit do fronty a postupně je zpracovávat. Fronta vláken je nastavena na výchozí hodnotu 25 vláken. Po zařazení vlákna do fronty třída zabezpečí postupné vykonávání vláken (je možné vlákno upřednostnit pomocí nastavení priority, jak jsme si již řekli v předchozím odstavci). Pokud je fronta plná, třída zařadí vlákno na konec a jakmile se uvolní místo, zařadí pozastavené vlákno do fronty čekajících požadavků.

Velikost fronty je omezena velikostí operační paměti, je důležité mít tento fakt na paměti, a to zvláště tehdy, pokud programujeme aplikaci v prostředí Compact frameworku, protože PDA přístroj obvykle není pamětí o dostatečné kapacitě k tomuto účelu vybaven.

5.2.4.3 Třída BackgroundWorker

Třída je dostupná ve vývoji desktop aplikací, tzn. pro naše nasazení na PDA bylo nutné její funkčnost naimplementovat pomocí již probrané třídy ThreadPool. Třída na rozdíl od samotné třídy ThreadPool umožňuje zrušit její celé provádění, tzn. všech vláken použitých k provádění operací na pozadí. Obsahuje i konstrukce pro informování o stavu prováděných operací, tzn. je možné definovat obslužnou funkci a informovat v ní uživatele, v jakém stavu se nachází prováděná operace – této funkčnosti budeme využívat například při stahování větších souborů ze vzdáleného serveru.

Implementace je provedena formou komponenty, tzn. třídu je možné přidat do projektu jakožto komponentu ve Visual Studio Designeru. Třída rovněž zpracovává výjimky za běhu částí aplikace v pozadí.

5.2.5 Popis implementace uživatelského rozhraní

Obrázek 5.5 znázorňuje obrazovku uživatele po spuštění aplikace, na které lze pomocí menu **Soubor** aplikaci vypnout nebo si zobrazit informace o aplikaci v menu **Nápověda**. Dole je k dispozici textová informace o právě prováděné akci.

Obrázek 5.5 – Úvodní obrazovka PDA

Na dalším obrázku, Obrázek 5.6 – Nabídka Soubor, je vidět dostupná nabídka – příkaz **Obnovit** explicitně spustí vyhledávání objektů v okolí.

Obrázek 5.6 – Nabídka Soubor

Příkaz **Zobrazit mapu** pak spustí aplikaci Google Maps pro Pocket PC, ve které se zobrazí na mapě seznam objektů. Mapu lze rovněž využít k prohledávání a orientaci v okolí. Pokud v aplikaci zapneme využití GPS, zobrazí se na ní modrý bod indikující současnou polohu přístroje. Červené balónky pak zobrazují uložené objekty v systému a po kliknutí na některého z nich se zobrazí základní informace, o jaký objekt se jedná. Modrý bod udávající aktuální polohu se postupně při pohybu uživatele s přístrojem na mapě posouvá, a jakmile se s přístrojem přiblížíme do okolí některého z bodů – tzn. bude nalezen nový objekt v okolí, aktivuje se naše aplikace a přenese se do popředí s upozorněním o nově nalezených objektech (zvukový, případně vibrační signál).

Obrázek 5.7 – Mapa

Seznam nově nalezených objektů je vidět na obrázku Obrázek 5.8 – Nově nalezené objekty. Uživatel je opět informován ve stavovém řádku aplikace a prohledávání okolí je pozastaveno.

Obrázek 5.8 – Nově nalezené objekty

Uživatel může pomocí seznamu (kliknutí na položku) vybrat objekt, nebo může využít hardwarových tlačítek na přístroji a pohybovat se tak po seznamu bez dotyku na obrazovku.

Pokud si uživatel nevybere objekt, který by ho zajímal, je možné znovu spustit prohledávání okolí zvolením položky **Pokračovat** v nabídce, jak je vidět na obrázku Obrázek 5.9 – Opětovné prohledávání okolí. Aplikace pak již nereaguje na objekty, které již byly zobrazeny v seznamu před aktivací nabídky **Pokračovat** novým upozorněním, že byly nalezeny nové objekty v okolí. Tyto objekty však v seznamu ponechá, dokud se vyskytují v okolí.

Obrázek 5.9 – Opětovné prohledávání okolí

Pokud se uživatel chce dozvědět o nějakém objektu bližší informace, může na název objektu v seznamu objektů kliknout a zobrazit si detail objektu. Klient po zvolení objektu ze seznamu naváže spojení se serverem a stáhne data do PDA.

Obrázek 5.10 – Detail objektu znázorňuje obrazovku s detailem objektu, na které se zobrazí, je-li zvolen, výchozí textový článek, v dolní části pak seznam dalších textových položek.

Obrázek 5.10 – Detail objektu

Ve spodní části aplikace jsou dále záložky **Audio** a **Video**, pomocí kterých je možné zobrazit seznam audio nebo video položek. Po jejich výběru systém opět navazuje spojení se serverem a aktualizuje data. Pomocí menu v dolní části obrazovky lze volbou položky **Zpět** uzavřít dialog detailu a vrátit se k seznamu okolních objektů, volbou položky **Zobrazení** lze zobrazit/schovat seznam textových položek. Výběrem možnosti **Nápověda** dojde k výpisu základních informací o aplikaci.

Obrázek 5.11 ukazuje stavový panel v místě záložek, kterým systém při každé akci informuje uživatele o své činnosti. Skrytí této možnosti lze provést kliknutím a podržením stylusu na místě, dokud se neobjeví kontextová nabídka a neumožní nám tuto akci provést. Nicméně ve většině případů se stavový panel skryje po úspěšně provedené akci sám.

Obrázek 5.11 – Možnost skrýt stavový dialog

Zobrazení nebo přehrání položky v seznamu nám umožní dvojí poklepání stylusem na položku. Textová data se zobrazují nahoře v prohlížeči, audio a video data zahájí stahování souboru a

informují uživatele o stavu stahování, a to formou množství procent (%) a velikostí (Kb/Mb) již stažené části ke celkové velikosti souboru. Stahování audio a video souborů je vzhledem k možné velikosti souboru zrušit. Tato možnost je uživateli nabídnuta po zahájení stahování v podobě změny hlavního menu dole na obrazovce. Obrázek 5.12 – Možnost zrušit probíhající stahování ukazuje zřetelně právě popsané vlastnosti. Implementace této funkce patřila mezi nejobtížnější, a to především z důvodu nutnosti běhu stahování v druhém procesu (aby bylo možné ovládat uživatelské rozhraní – zrušit stahování), .NET Compact Framework je právě o možnosti tohoto běhu – threadu oproti aplikacím na PC platformě poměrně ochuzen.

Obrázek 5.12 – Možnost zrušit probíhající stahování

Obrázek 5.13 – Aplikace čeká na uživatele znázorňuje informativní zprávu o aktuálním stavu systému a čekání na reakci uživatele. Jakmile se totiž soubor stáhne, spustí se v přehrávači a aplikace počká, dokud uživatel nebude chtít pokračovat v práci. Implementace této možnosti patřila mezi složitější z toho důvodu, že aplikace se na systému Pocket PC chová při kliknutí na křížek (tedy uzavření aplikace) poněkud jinak, než je uživatel z PC zvyklý. Aplikace se neuzavře, ale pouze přejde do pozadí (background) a stále běží. Tzn. po přehrání souboru přehrávačem Media Player a kliknutím na křížek aplikace přešla pouze do pozadí a přehrávaný soubor zůstal otevřený, tzn. nebylo možné jej smazat. Proto bylo nutné přehrávač speciálně spustit, poznamenat si jeho jedinečné id a poté ho i uzavřít.

Obrázek 5.13 – Aplikace čeká na uživatele

Co se týče nadpisů jednotlivých sloupců (dokonce i existence sloupců), tak tyto je možné měnit v databázi v uložených procedurách, jak bude probráno dále v sekci SQL serveru.

5.2.6 Instalace

Pro chod aplikace je nutné mít na PDA nainstalovaný .NET framework ve verzi 3.5 a vyšší. Pak již stačí zkopírovat spouštěcí soubor klientské aplikace na PDA a spustit (k tomuto se používá Microsoft ActiveSync), případně využít instalační aplikace.

Aplikace obsahuje konfigurační soubor .config, ve kterém se uchovává adresa webové služby, přes kterou probíhá komunikace s databází. Je ale tudíž nutné tento konfigurační řetězec změnit a nastavit ho na adresu serveru a následně virtuálního adresáře. Problematika bude prodiskutována v kapitole o webových službách. Při prvním spuštění aplikace, pokud ještě soubor .config neexistuje, aplikace nás požádá o konfigurační nastavení a toto nastavení si pak uloží.

5.2.6.1 Microsoft ActiveSync

Tato aplikace slouží ke komunikaci PDA s hostitelským počítačem a k synchronizaci (PDA je možné synchronizovat např. s poštou Microsoft Outlook, kalendářem, kontakty apod.).

Pomocí této aplikace lze přidávat a odebírat programy do/z PDA, případně kopírovat data apod. Po nainstalování nejnovější verze aplikace a připojení PDA k počítači, aplikace sama pozná připojené PDA a provede uživatele jednotlivými kroky k propojení PDA s PC. Poté je možné z nabídky programu vybrat průzkum zařízení a tím pádem na PDA přenést soubory, případně spustit instalaci, která se o to sama postará.

5.3 Server

Serverová část se skládá ze tří následujících částí:

- Administrační aplikace, která uživateli nabízí pohodlnou správu veškerých dat,
- SQL server,
- Web services – webové služby,
- Google maps – generování KML souboru.

Poslední dvě části, obě umístěné v IIS službě Microsoft Windows, budou prodiskutovány samostatně v dalších podkapitolách.

5.3.1 Popis implementace

Administrační část systému je navržena ke správě dat, tzn. k vedení informací o jednotlivých objektech. Uživatel může spravovat objekty, ke každému přidávat textová a multimediální data, případně již existující data měnit a mazat. Aplikace komunikuje s SQL serverem a samotné soubory ukládá na disk do složky s webovou službou (využívá k tomu podadresáře).

5.3.2 Google maps – generování KML souboru

Aplikace využívá celosvětové řešení map – http://maps.google.com, které jsou schopné zobrazit lokalizované mapy včetně satelitních snímků.

K zobrazování map využívá volně stažitelné aplikace Google Maps ve verzi Mobile (existují verze pro většinu mobilních přístrojů, včetně Pocket PC). Aplikace umožňuje využít i GPS zařízení, kdy na mapě je vidět i aktuální poloha přístroje na Zemi. Toho se dá využít k zobrazení mapy s objekty. Spolupráce spočívá v zobrazení souboru typu KML, tzn. vložení adresy se souborem do vyhledávání v aplikaci, přičemž aplikace si jej po prvním zadání pamatuje a není již nutné ji podruhé explicitně zadávat, dokud ji uživatel nezmění nebo nevyhledá jinou položku (i tak ale zůstává v posledně vyhledaných objektech – je ji možno lehce znovu aplikovat).

Součástí projektu je ASP.NET aplikace, která generuje KML soubor ve formátu, který je uveden v následujícím textu.

5.3.2.1 Popis KML souboru

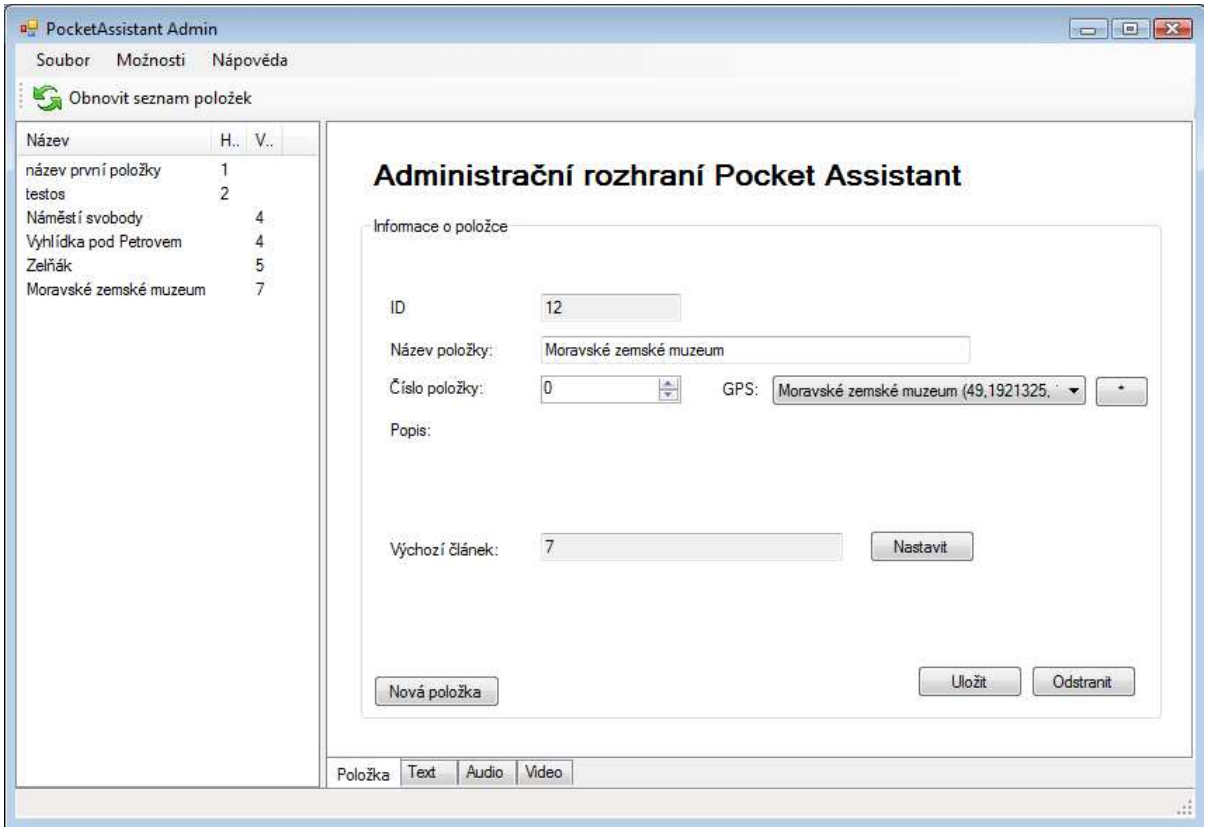
Souborem KML je možné popsat body na mapě. Jeho formát je definovaný v XML a v základní podobě vypadá následovně:

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.2">
  <Placemark>
    <name>Jméno bodu na mapě</name>
    <description>Popis bodu na mapě</description>
    <Point>
      <coordinates>16.6089402777777,49.19230722222222,0</coordinates>
    </Point>
  </Placemark>
  <Placemark>
    <name>Jméno druhého bodu na mapě</name>
    <description>Popis druhého bodu na mapě</description>
    <Point>
      <coordinates>16.6083333333333,49.195,0</coordinates>
    </Point>
  </Placemark>
</kml>
```

Sekce Placemark popisuje jeden bod na mapě. Name popisuje název bodu, Description popis bodu a sekce Point popisuje umístění bodu v koordinátách – coordinates v podobě longitude, latitude a volitelně altitude (v našem případě výškovou složku altitude nepotřebujeme).

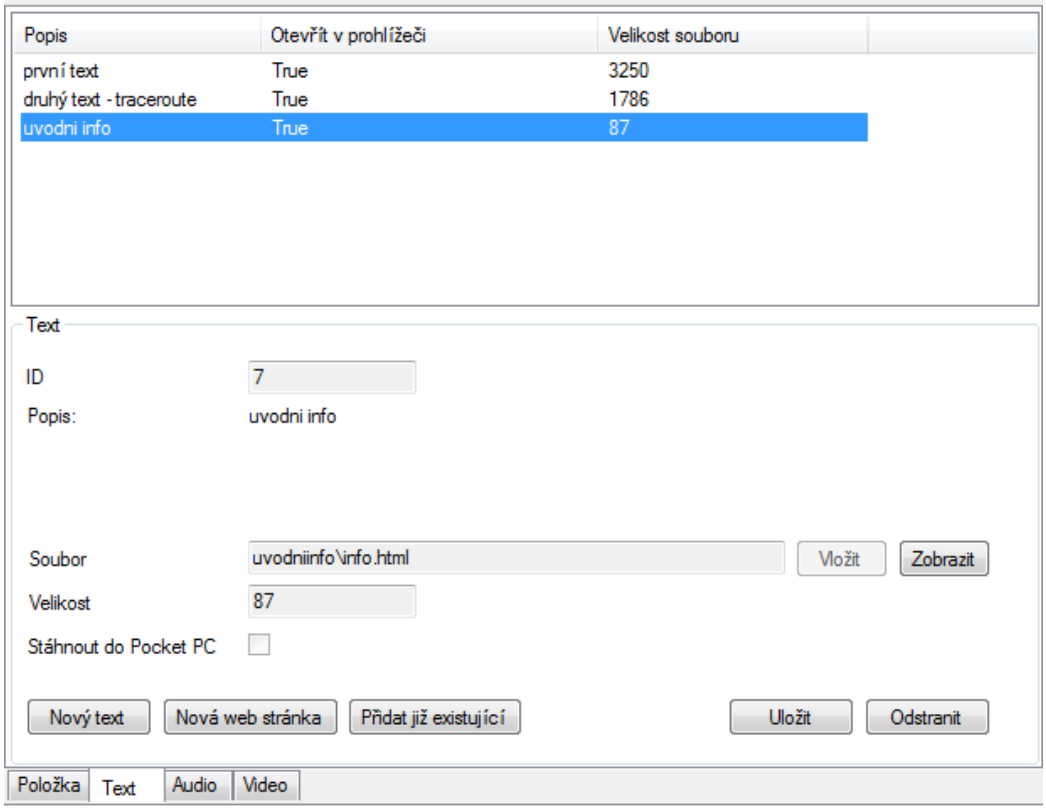
5.3.3 Popis implementace uživatelského rozhraní

Po spuštění aplikace je navázáno spojení s SQL databází pomocí přihlašovacího jména a hesla, které je možné změnit v nastavení programu. Po úspěšném navázání spojení se zobrazí seznam objektů, přičemž po poklepání na položku objektu v seznamu, se objekt načte do editačního pole vpravo a je možné jej editovat, případně odstranit. Obrázek 5.14 – Administrační uživatelské rozhraní ukazuje právě popsané možnosti.



Obrázek 5.14 – Administrační uživatelské rozhraní

Zároveň s načtením objektu do editačního pole se povolí přesun po záložkách umístěných dole v pravé části dialogu, čímž je umožněno spravovat textové, audio a video data jak je možno vidět na obrázku Obrázek 5.15 – Správa textových informací k objektu

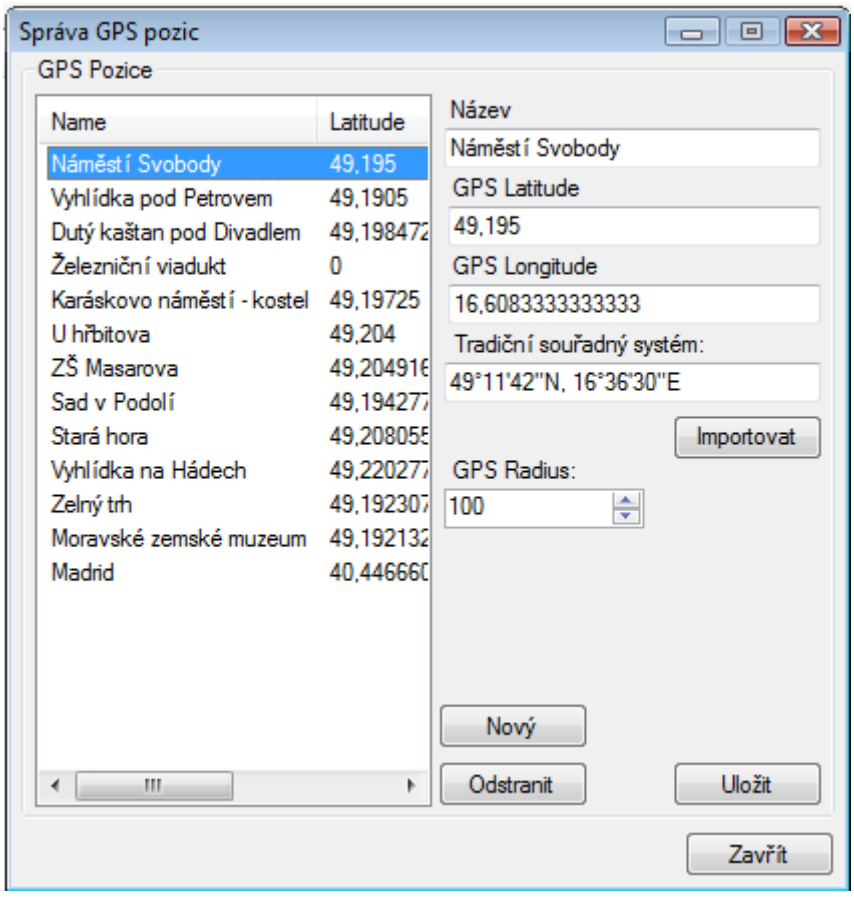


Obrázek 5.15 – Správa textových informací k objektu

Po přechodu na některou ze záložek se zobrazí seznam položek v systému a opět po poklepání na jednotlivou položku v seznamu lze dané položky editovat, případně smazat.

Aplikace dále nabízí možnost přidat již existující text v případě, že by některé objekty sdílely mezi sebou informace např. o společném autorovi.

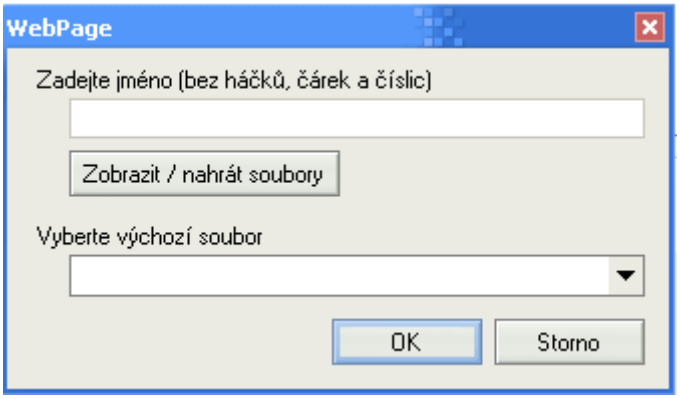
Ke každému objektu lze přidat GPS pozice z roletky. Samotné GPS pozice lze administrovat pomocí tlačítka s hvězdičkou vedle roletky s GPS pozicemi. Administrace GPS pozic je zobrazena na obrázku Obrázek 5.16 – Správa GPS pozic.



Obrázek 5.16 – Správa GPS pozic

Ke každé GPS pozici je možné přidat její jméno, latitude, longitude a radius, který vyjadřuje velikost okolí, které se má brát jako součást objektu. Navíc je možné převádět souřadnice mezi tradičním souřadným systémem, se kterým jsou zpravidla posluchači seznamováni na našich školách, a souřadným systémem složeným pouze z latitude a longitude, který je využíván v GPS zařízeních. Převod se jednoduše provede vložením řetězce s tradičním souřadným systémem do okénka a stisknutím tlačítka **Importovat**.

Obrázek 5.17 znázorňuje, jak lze u každé záložky (text, audio, video) přidat nový prvek, u textové záložky dále i novou webovou stránku. Po kliknutí na tuto volbu se otevře nový dialog, který vyzve k zadání jména (bez háčků a čárek). Toto jméno následně slouží k vytvoření složky, ve které budou uloženy soubory nově vytvářených stránek.



Obrázek 5.17 – Vložení jednoduché webové stránky

Následně po kliknutí na tlačítko **Zobrazit/nahrát** soubory se otevře složka, do které je možné nakopírovat obsah jednoduché webové stránky (*.html, *.htm, *.css, *.jpg, apod.). Po uzavření dialogu s obsahem složky je třeba vybrat výchozí soubor (obvykle index.html), což se provede kliknutím na vysouvací roletku. Po kliknutí na tlačítko **OK** se lze vrátir do administrační části, kde je možné zeditovat popis a novou webovou stránku uložit do databáze. Není možné ale zatrhnout možnost stažení souboru do PDA, protože k tomu je třeba dostupnost celé složky vystavené v adresáři na serveru.

Vkládání ostatních položek již probíhá jednoduše na základě editace jednotlivých položek k vyplnění a vybrání souboru, který chceme do databáze vložit. Po kliknutí na tlačítko **Vložit** a vybrání souboru se automaticky doplní jeho název a po uložení položky do databáze se doplní automaticky i jeho velikost a ID nově vložené položky. V té chvíli je umožněno i kliknutí na tlačítka spojená s prací nad uloženou položkou, např. její odstranění. Každá otevřená položka v systému má rovněž možnost zobrazení/přehrání, čímž se otevře vložená položka v programu asociovaném s danou příponou souboru.

Celý systém ukládání souborů a vytváření adresářů je založen na jménech vkládaných souborů/webových stránek. Pokud se vyskytne stejný název souboru, jaký se již v databázi nachází, tak jej aplikace sama přejmenuje (přidá na konec jména pomlčku a číslo), čímž je zajištěna možnost ukládat do databáze soubory a adresáře stejného jména. Do databáze se pak uloží soubor s již takto změněným jménem, aniž by tímto uživatel byl obtěžován a vyrušován.

Uživatelské rozhraní je koncipováno tak, že si je může uživatel přizpůsobit podle svých představ, samozřejmě s ohledem na možnosti systému a zařízení. Jak výpis objektů vlevo, tak i pravá část je upravitelná co do velikosti oblastí jednotlivých oblastí. Tzn. je možné roztáhnout výpis objektů např. přes půl obrazovky, nebo si lze roztáhnout výpis samotných položek v záložce textu. Celá koncepce je založená na co největším komfortu uživatele s prací v daném prostředí.

5.3.4 Administrace GPS pozic na Pocket PC zařízení

Obrázek 5.18 – Administrace GPS pozic na Pocket PC zobrazuje správu GPS pozic na samotném zařízení. Administrátor systému takto může snadno zadat přesné pozice do systému přímo v „terénu“. Má rovněž možnost stávající pozice upravit, případně smazat.

Pokud přistroj má aktivní GPS zařízení, je možné doplnit koordináty přímo ze zařízení a tak velice jednoduše snímat polohy objektů přímo na místě.

Obrázek 5.18 – Administrace GPS pozic na Pocket PC

Aplikace po spuštění naváže spojení se vzdáleným serverem za pomocí webové služby, stáhne seznam GPS pozic a čeká na uživatele. Uživatel pak může přidat novou pozici, stávající pozice upravit nebo odstranit. Sejmutí aktuální GPS pozice je možné provést stisknutím tlačítka **Získat z GPS**.

5.3.5 SQL server

Celý systém využívá služeb nainstalovaného SQL serveru v aktuální verzi 2005, ve kterém musí být vytvořena struktura pro uložení dat a tabulek diskutovaných výše. Systém pracuje na základě spouštění uložených procedur v SQL serveru a na prezentaci jejich výsledků aplikaci. Tzn. tímto je zajištěna nezávislost na struktuře databáze a se zachováním vstupních a návratových parametrů je

možné si strukturu databáze pro uložení dat s patřičnými znalostmi jakkoliv upravit. Následně je pro ilustraci uveden jeden z použitých skriptů uložené procedury pro smazání textové položky z databáze:

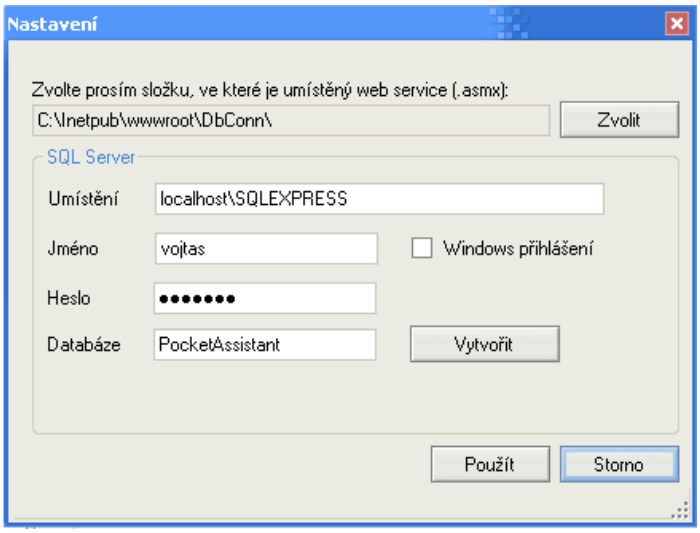
```
-- vraci -1 pokud polozka neexistuje
-- vraci -5 pokud je text jeste nekde v databazi
-- vraci 0 pri OK
CREATE PROCEDURE [dbo].[DeleteText]
@ID int,
@Value int,
@Filename varchar(1024) out,
@Return int output
AS
BEGIN
	DECLARE @cnt int
	DECLARE @Item_ID int

	SELECT @cnt = COUNT(*) FROM Text WHERE ID = @ID;
	IF @cnt = 0
		BEGIN
			SELECT @Return = -1
			return
		END

	SELECT @Item_ID = Item_ID FROM Item WHERE Value = @Value
	SELECT @Filename = Filename FROM Text WHERE ID = @ID

	DELETE FROM br_item_text
	WHERE Text_ID = @ID AND Item_ID = @Item_ID

	SELECT @cnt = COUNT(*) FROM br_item_text WHERE Text_ID = @ID;
	IF @cnt > 0
		BEGIN
			SELECT @Return = -5
			return
		END

	DELETE
	FROM Text
	WHERE ID = @ID;

	SELECT @Return = 0
END
```

Systém uložených procedur pracuje na způsobu návratových hodnot informujících o výsledku provedené akce. Tímto způsobem je možné zajistit v programu, jestli při provedené akci nedošlo k chybě a pokud ano, tak k jaké. Každá uložená procedura vrací parametr @Return, který udává výsledek operace, a je tedy možné oznámit, že požadovaná textová položka neexistuje, případně ve výše uvedené ukázce dáváme na vědomí, že textová položka figuruje ještě u jiného objektu v databázi, tzn. není možné smazat záznam v databázi a soubor na disku představující tuto textovou položku.

5.3.5.1 Instalace

SQL Server 2005 Express Edition je možné stáhnout z webové adresy společnosti Microsoft http://www.microsoft.com/sql/editions/express/default.mspx (datum přístupu 25. 3. 2005).

Poté co se ji podaří nainstalovat do systému, je třeba vytvořit schéma databáze a uložené procedury pomocí SQL skriptu dodaného s vyvíjenou aplikací v souboru Install.sql. Následně je třeba vytvořit dalšího uživatele s přístupem k vytvořené databázi, jeho jméno a heslo (login a password) zadat v dialogu nastavení v administrační části systému spolu s adresou SQL serveru. Obrázek 5.19 ukazuje umístění na počítači 608-27AHXA a instance se jménem SQLEXPRESS (pokud při instalaci neuvedete jinak a nemáte už nějaký SQL server nainstalovaný, tak by měl být vždy tohoto názvu – lze zjistit v nastavení SQL serveru), vytvořený uživatel pro přístup k databázi má jméno vojtas a zvolené heslo. Dále je uveden název databáze (pokud jej nezměníte, měl by být rovněž stejný jako v ukázce). Následně je nutné zvolit složku, ve které se nachází webová služba. Ze strany administračního rozhraní je také možné vytvořit toto vše automaticky, a to kliknutím na tlačítko **Vytvořit** u jména databáze. Aplikace se uživatele následně zeptá na jméno a heslo nového uživatele a pak vytvoří strukturu databáze i s uloženými procedurami a údaje doplní do dialogu automaticky. Rovněž se uživatele zeptá, zda chce tyto údaje uložit do souboru s webovým servisem. K této funkci nutné přihlášení uživatele pomocí Windows účtu, který musí mít administrátorská práva s nainstalovaným SQL Server Express ve výchozím nastavení (tak jak je dodáván s aplikací), což pokud je SQL server správně nainstalován, je splněno.



Obrázek 5.19 – Nastavení v administrační části

5.3.6 Webové služby

Anglický název Web services by se dal česky přeložit jako webové služby, což není zcela přesné, protože dnes se nejedná jen o služby poskytované přes web. Webová služba je samostatná modulární aplikace, která může být umístěna kdekoliv na webu, kde je nainstalovaný a zprovozněný IIS (Internet Information Services – Internetové Informační Služby) od společnosti Microsoft. Funkcionálně může být velice triviální (pouze předávání cen akcii, viz. např. finanční část portálu společnosti YAHOO, který takto pracuje s cenami na burzách), ale také komplexní (např. sledování obrazu z webové kamery apod.).

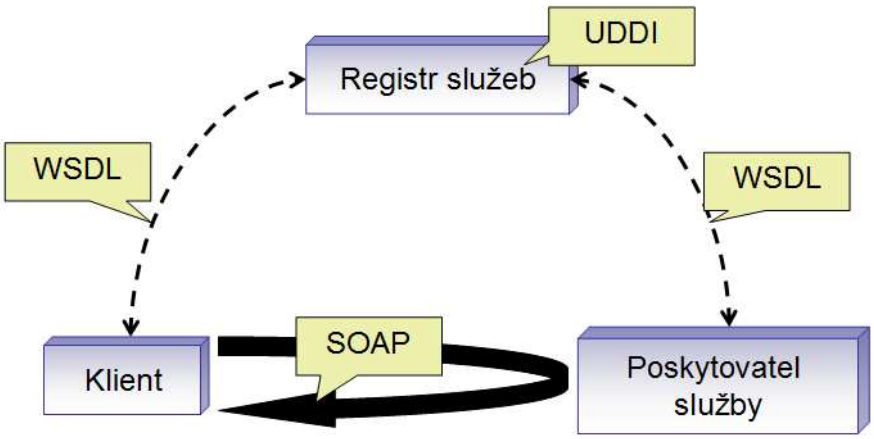
Z technického pohledu používají webové služby koncept klient-server. Klient si se serverem vyměňuje zprávy ve většině případů pomocí webového protokolu SOAP uložené ve formátu XML, jejichž tvar je definován pomocí WSDL (Web Service Description Language) a registrován v UDDI (Universal Discovery Description and Integration), čímž ho potencionální uživatelé mohou snadno najít. Server může být distribuován kdekoliv na internetu, ale stejně tak je možné jej umístit kdekoliv na interní síti podniku. Navíc tím, jak již bylo zmíněno, komunikuje zprávami ve formátu XML, proto je možné zpracovat výsledky operací v různých vývojových jazycích a na různých platformách. Protože dále webové služby pracují se standardními webovými protokoly – XML, HTTP a TCP/IP, což je rovněž výrazná výhoda, neboť pak je možné využít jeho služeb i např. přes proxy server, což u normální klient-server aplikace není tak jednoduše možné. Navíc na základě popisu WSDL existují nástroje jako např. Visual Studio, a ty jsou schopné vygenerovat tzv. proxy třídy, které jsou používány klientskými aplikacemi pro komunikaci s webovou službou.

Webové služby fungují na principu vzdáleného volání funkcí v distribuovaných systémech pomocí RPC (Remote Procedure Call), jenž je popsána v SOAP protokolu. SOAP protokol popisuje, jak má vypadat zpráva posílaná od klienta serveru s volanou funkcí, parametry předané funkci a následně server vrátí zprávu s výsledky volané funkce. Další volitelnou možností SOAP specifikace je, jak má vypadat HTTP hlavička obsahující SOAP zprávu.

WSDL soubor je XML soubor popisující SOAP zprávy jako takové a způsob jejich obměny. Používá se k popisu formátu zpráv založených na XML standardu, což znamená, že jsou oba nezávislé na programovacím jazyce a je možné je zpracovat na různých platformách. WSDL rovněž definuje kde je servis dostupný a jaké komunikační protokoly je možné použít ke komunikaci s ním. Ve zkratce lze uvést, že WSDL soubor definuje vše, co potřebujete ke psaní programu, který využívá danou webovou službu. Existuje několik nástrojů ke generování WSDL souboru z existujícího zdrojového kódu. Microsoft Visual Studio .NET přímo tento nástroj obsahuje, takže vygeneruje toto schéma za nás a my na něj pouze odkážeme v aplikaci, kde ho chceme využívat.

UDDI neboli Universal Discovery Description and Integration jsou takové „Zlaté stránky“ webových služeb. Můžete hledat společnost, která poskytuje servis, který potřebujete, vyhledat nějaké

informace potřebné ke kontaktování někoho odpovědného pro získání dalších informací. UUDI záznam je XML soubor, který popisuje společnost a služby, které nabízí. Lze rovněž vyhledávat podle geografické polohy a získávat informace, kontakty, odkazy a technická data, např. ke zjištění, který ze servisů vyhoví zadaným požadavkům. Na obrázku Obrázek 5.20 – Schéma architektury webových služeb je vidět přehledně základní schéma této architektury.



Obrázek 5.20 – Schéma architektury webových služeb

Využití webových služeb pro implementaci bylo zvoleno právě z výše uvedených důvodů. Je tak možné nejenom využít univerzálnosti tohoto nástroje, ale získat tak i oddělenou vrstvu, kterou lze měnit nezávisle na klientovi – PDA. Nesmí být rovněž opomenuta možnost vývoje klienta na jiném jazyku a platformě – díky webové službě je možné koncept celého systému využít i například v jazyku Java, otevírá se možnost naprogramovat i klienta pro mobilní telefony. Navíc díky tomu, jak již bylo uvedeno, je možné projít přes firewally, proxy servery apod., což by přes klasickou klient-server architekturu nebylo jednoduše možné (port 80, přes který se provozují http požadavky, nebývá blokován – na rozdíl od ostatních portů). To vše je možné právě z toho důvodu, že webová služba je realizována na platformě Microsoft jako speciální druh ASP.NET aplikace a právě díky silné podpoře v ASP.NET runtime a v .NET frameworku je programátor oproštěn od psaní velmi složité infrastruktury kódu.

Jednoduše se dá komunikace popsat několika kroky:

- Klient dynamicky založí objekt proxy a vyvolá jeho metodu. Metoda má stejné jméno, parametry i návratovou hodnotu jako metoda implementovaná na straně webové služby.
- Proxy serializuje parametry do souboru XML a následně vytvoří SOAP dokument. Ten pošle webovému serveru přes protokol HTTP.
- ASP .NET pozná, že jde o dotaz na web službu. Podle obsahu SOAP dokumentu spustí odpovídající metodu webové služby. Data z dokumentu jsou deserializována a předána jako parametry.
- Metoda vrátí výsledek a ASP .NET runtime se postará o zpětnou serializaci odpovědi opět ve formě souboru XML.
- Odpověď je přijata objektem proxy. Jeho úkolem je deserializovat získaná data a předat je zpět volajícímu kódu.
- Kód klientské aplikace dále pokračuje ve vykonávání příkazu. Toto vše platí pro synchronní komunikaci, v případě asynchronní komunikace program nečeká na vrácení výsledku provedené operace, ale pokračuje dál.

5.3.6.1 Ukázka návratového XML webové služby

Vyvolání funkce pro seznam GPS souřadnic v systému navrátí XML soubor v následujícím tvaru:

```
<?xml version="1.0" encoding="utf-8"?>
<DataSet xmlns="http://PocketAssistantWebService.net/">
  <xs:schema id="NewDataSet" xmlns=""
xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:msdata="urn:schemas-
microsoft-com:xml-msdata">
    <xs:element name="NewDataSet" msdata:IsDataSet="true"
msdata:UseCurrentLocale="true">
      <xs:complexType>
        <xs:choice minOccurs="0" maxOccurs="unbounded">
          <xs:element name="Table">
            <xs:complexType>
              <xs:sequence>
                <xs:element name="ID" type="xs:int" />
                <xs:element name="Name" type="xs:string" />
                <xs:element name="Latitude" type="xs:double" />
                <xs:element name="Longitude" type="xs:double" />
                <xs:element name="Radius" type="xs:double" />
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        </xs:choice>
      </xs:complexType>
    </xs:element>
  </xs:schema>
```

```
  <diffgr:diffgram xmlns:msdata="urn:schemas-microsoft-com:xml-msdata"
xmlns:diffgr="urn:schemas-microsoft-com:xml-diffgram-v1">
    <NewDataSet xmlns="">
      <Table diffgr:id="Table1" msdata:rowOrder="0">
        <ID>17</ID>
        <Name>Náměstí Svobody</Name>
        <Latitude>49.195</Latitude>
        <Longitude>16.6083333333333</Longitude>
        <Radius>100</Radius>
      </Table>
      <Table diffgr:id="Table2" msdata:rowOrder="1">
        <ID>18</ID>
        <Name>Vyhlídka pod Petrovem</Name>
        <Latitude>49.1905</Latitude>
        <Longitude>16.6085</Longitude>
        <Radius>90</Radius>
      </Table>
    </NewDataSet>
  </diffgr:diffgram>
</DataSet>
```

5.3.6.2 Instalace

Pro instalaci webové služby je nutné mít nainstalovaný IIS (internetové informační služby) v systému (není podporováno ve všech operačních systémech společnosti Microsoft). Následně je pak třeba vytvořit nový virtuální adresář a umístit tam soubory potřebné k chodu webového servisu (verzi ASP.NET pro virtuální adresář je nutné nastavit na verzi 2.0).

Dále je nutné vytvořit adresář, do kterého se webová služba umístí, cestu k němu pak zadat do administrační části systému. Obrázek 5.19 znázorňuje možnost tohoto zápisu. Zvolenou cestu nastavíme v konfiguraci klienta na PDA, viz. předchozí kapitola 5.2.6. Správný chod webové služby (tedy správnou konfiguraci) je možné ověřit zadáním adresy http://localhost/ + virtuální adresář + /Service.asmx, kdy by se nám měla zobrazit stránka obsahující výpis jeho funkcí.

5.3.6.3 Instalace podpory Google Maps

V rámci instalace webové služby se nainstaluje i podpora pro mapové řešení společnosti Google. Správný chod mapové služby (tedy správnou konfiguraci) je možné ověřit zadáním adresy http://localhost/ + virtuální adresář + /KML.aspx. Její aplikace ve webové stránce je pak vidět v aplikaci http://localhost/ + virtuální adresář + /Map.aspx. Malou komplikací v tomto případě je nutná změna kódu pro vložení mapy – společnost Google vyžaduje registraci pro každý webový portál, ale protože mapové řešení Map.aspx není vyžadováno pro správný běh aplikace, tzn. může se zprovoznit pouze v případě potřeby.

6 Další rozvoj aplikace

6.1 Připojení k web službě

Do budoucna by bylo vhodné rozšířit webovou službu o bezpečnostní prvky, tzn. po připojení ke službě by se musel klient autentifikovat, a to například na základě jednoznačného ID kapesního přístroje.

6.2 Vývoj klientské části na platformě Java ME

Vývoj aplikace pro mobilní telefon, zajištění komunikace s GPS a následná spolupráce s existující webovou službou.

6.3 Automatická instalace na server

Instalace programu na počítač, včetně SQL serveru s výchozím nastavením a schématu databáze s nastavením uživatele pro přístup. Následně i instalace IIS služeb a samotné webové služby. To vše je částečně splněno samotnou programovou částí, kromě instalace IIS.

6.4 Ukládání konfigurace

Implementovat možnost ukládání stavu grafického rozhraní do souboru na disk. Tzn. pokud uživatel zvětší velikost okna apod., uchovala by se tato změna i pro příští spuštění aplikace.

6.5 Chybové zprávy

Přidat vrstvu pro překlad nejčastějších chybových zpráv samotného SQL serveru, se kterými se může uživatel setkat.

6.6 Nápověda

Vytvoření základní nápovědy jak pro administrační rozhraní, tak pro klienta na PDA.

7 Závěr

Rozvoj internetu a s tím souvisejících webových služeb posouvá oblast vývoje aplikací opět o něco dále. Nic nám nebrání umístit webovou službu na server a nabídnout její funkčnost mnohým aplikacím a rozšířit tak svou působnost a nabídnout tím zákazníkům více než konkurence.

Ve spojení s určením polohy na Zemi se naskýtají nepřeberné možnosti vývoje aplikací. Jednou z nich může být využití právě této aplikace. Cílem bylo vyvinutí „elektronického průvodce“ pro uživatele, který má zájem o individuální procházky různými zajímavými lokalitami, a to i bez průvodce, kdy se může v klidu seznamovat s tím, co ho aktuálně nejvíce zajímá bez strachu, že se v dané lokalitě ztratí. Je to samozřejmě pouze jedna z možností využití nových technologií, které nám nový trend pod značkou GPS nabízí. Nicméně může být pro uživatele velmi zajímavá a přínosná.

Aplikaci lze dále rozšiřovat, doplňovat a vylepšovat, tak jak bylo zmíněno také v předchozí kapitole. Například díky zavedené koncepci webové služby a rozvoji mobilních telefonů s integrovaným GPS modulem je možné jednoduše implementovat klientskou část např. v jazyce Java a zpřístupnit tak veškerou funkčnost systému i do této oblasti. Pak již nebude nic snazšího než aplikaci stáhnout, nainstalovat, a udělit uživateli přístup do systému - kterýkoliv uživatel s takto vybaveným přístrojem bude schopen využít veškerého jeho potenciálu s minimem nákladů a snahy.

Jsem rád, že se mi podařilo vyzkoušet další oblast vývoje, a to mobilních zařízení ve spolupráci s databázovým serverem a webovou službou. Práce obohatila teoreticky i prakticky moje dosavadní vědomosti a zkušenosti v nové oblasti, do které se mi realizací této aplikace podařilo hlouběji proniknout. Získal jsem nové praktické poznatky při vývoji pod platformou .NET s využitím webových aplikací a mobilních aplikací při implementaci vícevláknového běhu úloh. Zajímavá byla i část práce, kde jsem se mohl seznámit s prací a využitím technologie XML. Přínosem mi byla i možnost vývoje aplikace se střední vrstvou a tzv. „byznys“ logikou, v dnešní době poměrně populární metodou vývoje aplikací podobného charakteru. Tyto metody řešení umožňující oddělení aplikace běžící na PDA od přímého přístupu k databázi byly rovněž pozitivem, neboť pak je možné bez úprav aplikace na PDA měnit zpracování požadavků a plnit s nimi poměrně jednoduše naše požadavky (změna zdroje dat apod.).

Celkově mohu říci, že práce splnila moje představy. Při jejím vývoji jsem si osvojil nové metody, přístupy a technologie v oblasti, která v současné době nabývá velkého významu. Domnívám se, že technologie založené na spolupráci s GPS zařízením budou i nadále na vzestupu a mohu tedy reálně předpokládat, že řadu právě nabytých znalostí využiji i při svém dalším profesním zaměření.

Literatura

[1] Albahari J., Albahari B.: C# 3.0 in a Nutshell, Third Edition, O'Reilly & Associates 2007.

[2] DePriest, D. NMEA data, dokument www dostupný na adrese http://www.gpsinformation.org/dale/nmea.htm, [přístup 26. 2. 2008].

[3] Dana, P. Global Positioning System Overview, University of Texas, 1994, dokument dostupný na adrese http://www.colorado.edu/geography/gcraft/notes/gps/gps_f.html, [přístup 26. 2. 2008].

[4] Fergus, D., Roof L.: The Definitive Guide to the .NET Compact Framework, Apress 2003.

[5] Fox, D., Box J.: Building Solutions with the Microsoft .NET Compact Framework: Architecture and Best Practices for Mobile Development, Addison Wesley 2003.

[6] Hasan, J., Duran M.: Expert Service-Oriented Architecture in C# 2005, Second Edition, Apress 2006.

[7] Horner M.: Pro .NET 2.0 Code and Design Standards in C# (Pro), Apress 2005.

[8] KML Tutoriál, dokument www dostupný na adrese http://code.google.com/apis/kml/documentation/kml_tut.html, [přístup 26. 2. 2008].

[9] Lee, W., Jepson B.: Programming the .NET Compact Framework, O'Reilly & Associates 2008.

[10] Nagel, Ch.: Enterprise Services with the .NET Framework: Developing Distributed Business Solutions with .NET Enterprise Services, Addison Wesley 2005.

[11] Price J.: Mastering C# Database Programming, Sybex 2003.

[12] Thompson, D., Miles, R.: Embedded Programming with the Microsoft .NET Micro Framework, Microsoft Press 2007.

[13] Tiffany R.: SQL Server CE Database Development with the .NET Compact Framework, Apress 2003.

[14] Troelsen, A.: Pro C# 2008 and the .NET 3.5 Platform, Apress 2007.

[15] UML, dokument www dostupný na adrese http://cs.wikipedia.org/wiki/UML, [přístup 26. 2. 2008].

[16] Wigley, A.: Microsoft .NET Compact Framework, Microsoft Press 2003.

[17] Yao, P., Durant, D.: .NET Compact Framework Programming with Visual Basic .NET, Addison Wesley 2004.

Seznam příloh

Příloha 1. CD/DVD se zdrojovými kódy, elektronickou podobou práce a její prezentací.