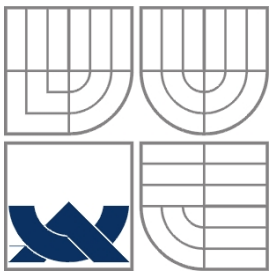
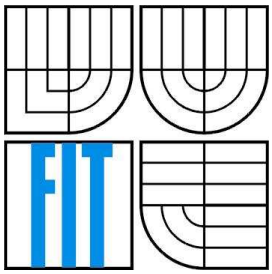


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

SLEDOVÁNÍ POLOHY S VYUŽITÍM GPS A PDA

POSITION TRACKING WITH GPS AND PDA

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

BC. VILÉM ČERNOHORSKÝ

VEDOUCÍ PRÁCE
SUPERVISOR

ING. IGOR SZÖKE

BRNO 2008

Abstrakt

Tento dokument vede k vytvoření aplikace pro vzdálené sledování polohy. Nejprve seznamuje čtenáře s historií určování geografické polohy a navigace. Ukazuje vývoj navigace přes primitivní historické pomůcky až k radiovým navigačním systémům. Podrobně popisuje princip a možnosti globálního pozičního systému GPS včetně protokolu NMEA, který využívají jeho přijímače. Jsou zde představeny současné navigační programy. V závěru dokumentu je provedena analýza, objektový návrh a implementace systému pro vzdálené sledování polohy pomocí GPS a PDA.

Klíčová slova

Geografická poloha, navigace, kompas, buzola, sextant, gyroskop, Loran, Omega, Transit, Glonas, Galileo, GPS, NMEA, TomTom, iGO, Route 66, AutoRoute, PDA, Pocket PC, Windows Mobile, API, Windows Sockets, RS-232, null-modem.

Abstract

This work presents steps in creation of remote position tracking application. Document introduces history of geographical positioning and navigation, and describes development of navigation from primitive utilities to complex radio navigation systems. Global Positioning System is described in more detail including NMEA protocol used by GPS receivers. This work also presents several current navigation applications. Based on obtained information work presents analysis, object-oriented design and implementation of remote position tracking system.

Keywords

Geographical position, navigation, compass, sextant, gyroscope, Loran, Omega, Transit, Glonas, Galileo, GPS, NMEA, TomTom, iGO, Route 66, AutoRoute, PDA, Pocket PC, Windows Mobile, API, Windows Sockets, RS-232, null-modem.

Citace

Černohorský Vilém: Sledování polohy s využitím GPS a PDA. Brno, 2008, diplomová práce, FIT VUT v Brně.

Sledování polohy s využitím GPS a PDA

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením Ing. Igora Szökeho.
Další informace mi poskytl Ing. Jaroslav Kadlec.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Vilém Černohorský
21. ledna 2008

Poděkování

Na tomto místě bych rád poděkoval svému vedoucímu, Ing. Igoru Szökemu, za pečlivou kontrolu textů a věcné připomínky k řešenému projektu.

Obsah

1 Úvod

Snaha člověka o orientaci v prostoru je známa už od nejstarších dob. Identifikace, pamatování si objektů a pozemních značek jako orientačních bodů byly metody, které používal již pračlověk při hledání cesty přes džungli nebo poušť. Prvními navigačními pomůckami tak byly kameny položené na cestě, značky na stromech a hory. Představovaly první příklady orientačních bodů, jak je známe dnes. Jakmile člověk začal zkoumat oceán, byl postaven před nový problém. Identifikace známých bodů byla snadná na zemi, na moři však jedinými viditelnými objekty byly slunce, měsíc a hvězdy. Začala tedy éra nebeské navigace, která byla prvním opravdovým řešením, jak se orientovat v neznámém prostoru. Vzájemná poloha hvězd a jejich geometrické uspořádání je v různých částech Země jiné. Díky tomu je pozorováním konfigurace hvězd možné odhadnout svou polohu na Zemi a směr k cíli. Geometrická konfigurace hvězd byla později určena pomocí měření úhlu mezi jednotlivými objekty na obloze. Byly vyvinuty přesné optické přístroje pro měření těchto úhlů a vytvořeny grafy a mapy, které usnadňovaly obtížnou výpočetní úlohu. I přesto byl postup měření úhlů mezi hvězdami velmi zdlouhavý a nepřesný. Navíc nebylo možné použít tuto metodu během dne nebo při zatažené obloze v noci. Naměřené úhly musely být zakresleny jako speciální grafy, ze kterých pak byla náročným výpočtem určena poloha s přesností několika kilometrů. Výpočetním postupem byla základní triangulační geometrie. Hvězdy se staly vrcholem trojúhelníku, měřené úhly mezi nimi umožňovaly navigátorovi řešit prvky trojúhelníku a určit hledanou polohu. Triangulace by byla mnohem snadnější a efektivnější, kdyby se daly měřit také vzdálenosti ke hvězdám. Tyto vzdálenosti by byly použity pro řešení prvků trojúhelníku namísto úhlů. Bohužel v té době taková měření nebyla možná.

Nápad automatického výpočtu polohy pomocí měření vzdálenosti k známým orientačním bodům dospěl k realizaci teprve v nedávné době. Využití radiových vln ve službách navigace znamenalo přelom v tomto oboru. Navigace byla konečně dostatečně přesná a určení polohy bylo zajišťováno automaticky strojem. Nebylo však jednoduché globálně pokrýt celou planetu, navíc navigační systémy byly dostupné pouze armádám, či jiným složkám, nikoli však obyvatelstvu. Vrcholem vývoje a současným stavem navigačních systému byl návrat do historie, konkrétně do éry nebeské navigace. Dnes již není potřeba měřit úhly mezi hvězdami. Na oběžnou dráhu naší planety byly vyneseny umělé satelity, které nám nebeská tělesa nahrazují. Jsou pro nás viditelné ve dne i v noci, za jakéhokoli počasí. Navíc díky radiovým signálům, které vysílají, lze určit jejich vzdálenost. Navigace se v posledních letech stala konečně dostupnou. Dnes již není její použití předurčeno pouze armádě, letectví a lodní dopravě. Navigační systémy využívají obyčejní lidé například při sportu, rekreaci nebo při každodenní cestě svým automobilem do práce. Navigace je dnes rychlá, dostatečně přesná a cenově dostupná záležitost. Ten nejjednodušší navigační systém lze vměstnat i do náramkových hodinek.

Tento dokument se snaží čtenáři přiblížit analýzu, návrh a implementaci programového vybavení, jež bude za pomoci dostupných navigačních systémů a bezdrátových datových sítí umožňovat sledování geografické polohy uživatele na vzdáleném počítači. Tento systém najde uplatnění zejména v dopravní infrastruktuře, u velkých spedičních firem a podobně, kde je zapotřebí monitorovat polohu služebních vozidel a případně operativně měnit trasu jejich cesty. Vzdálené sledování polohy není zcela novým problémem na poli navigace. Existuje již několik řešení, které však nejsou natolik univerzální, aby jejich nasazení bylo realizovatelné v širokém spektru uživatelů. Proto si tato práce dává za cíl, nabídnout uživateli systém, který je závislý pouze na celosvětově dostupných prostředcích, kterými jsou globální poziční systém GPS a datové sítě mobilních operátorů. Tento předpoklad nelimituje nasazení aplikace pouze do velkých firem, ale předurčuje ji k použití i u běžných uživatelů.

Následující kapitola popisuje současné možnosti určování geografické polohy a navigace. Představuje primitivní navigační prostředky a způsob jejich použití. Následně je popsán objev měření vzdálenosti pomocí radiových signálů a jeho využití pro navigaci. Kapitola 2.1 popisuje chronologicky nejprve pozemní navigační systémy. Poté se zaměřuje na dnes dostupné družicové navigační systémy, zkoumá jejich výhody a nevýhody, provádí základní srovnání a poodhaluje jejich budoucí vývoj. V následující kapitole je podrobně představen poziční systém GPS, který byl zvolen jako nejvhodnější řešení pro implementaci projektu. Je zde představena jeho historie, vysvětlen princip činnosti systému a způsob kódování vysílaných signálů. Následuje podkapitola o protokolu NMEA, pomocí kterého GPS přijímače komunikují s okolím. Nechybí zde popis vlastností a struktury tohoto standardu. V poslední podkapitole je shrnutí aplikačních řešení navigace s důrazem na přiblížení jejich možností vzdáleného sledování polohy. Kapitola 3 popisuje analýzu a návrh řešeného problému. Na začátku jsou uvažovány potřeby uživatele a stanoveny požadavky kladené na aplikaci. Následuje výběr implementačního prostředku a vlastní návrh aplikace. Návrh je veden od celků k detailům. Kapitola 4 popisuje realizaci projektu. Jsou zde zmíněny použité prostředky a nástroje a jejich výběr je zdůvodněn. Dále je vysvětlen pojem událostmi řízeného programování a je popsána základní struktura aplikace pro platformu PC a Pocket PC s operačním systémem Microsoft Windows. Další odstavce této kapitoly popisují vlastní implementaci projektu a přidružených tříd. Předposlední kapitola se zabývá testováním vytvořených aplikací a celého systému. Závěr popisuje dosažené výsledky. Naznačuje jak bude vývoj aplikace pokračovat a hodnotí do jaké míry bylo splněno zadání projektu.

Obsah tohoto dokumentu a práce na projektu navazuje na stejnojmenný Semestrální projekt. V jeho rámci byly nastudovány možnosti navigačních systémů a byla provedena analýza a návrh aplikace, která umožňuje sledování vzdáleného projektu. Získané poznatky a objektový model aplikace byly využity pro samotnou realizaci projektu.

2 Určování geografické polohy

Určování geografické polohy a s tím související pojem navigace je snaha, jak kdekoliv na Zemi stanovit svoji polohu a nalézt nejkratší cestu ke svému cíli. Metod jak stanovit geografickou polohu objektu je několik. Dá se použít takzvaná srovnávací navigace, při které se porovnává terén okolí objektu nebo osoby s mapou. Zde se dá využít i matematických výpočtů, například triangulace[1]. Triangulace se obvykle používá pro účely navigace, geodézie, metrologie a astrometrie.

Polohu a směr cesty je také možno stanovit terestricky pomocí kompasu, buzoly nebo gyroskopu. Kompas je zařízení určující světové strany. Typický kompas se skládá z volně pohyblivé magnetické střelky tvaru štíhlého kosočtverce, umístěné v pouzdře, na jehož dně je nakreslena kompasová růžice. Tato střelka si zachovává směr sever-jih podle zemského magnetického pole. Dnes existují i kompasy založené na mnoha dalších fyzikálních principech. Prvním kompasem byl nejspíše kus magnetovce, který byl položený na dřevěné podložce v nádobě s vodou. Klasický kompas se používal ve středověku a novověku hlavně v námořní dopravě. Zdokonalenou verzí kompasu, která používá azimut, je buzola. Buzola je jednoduchý přístroj, který slouží k orientaci v terénu, určování světových stran a měření azimutu. Základem buzoly je kompas, který svou střelkou ukazuje na magnetický pól Země. Pro běžné účely orientace je to dostatečně přesný směr k severnímu zeměpisnému pólu Země. V případě potřeby přesného měření je nutné vzít v úvahu magnetickou deklinaci Země a měření korigovat. Buzola je také opatřena průzorem, kterým se nastavuje směr k bodu v terénu. Měření se provádí pomocí otočného kotouče se stupnicí s vyznačenými úhly. Světové strany jsou označeny počátečními písmeny jejich anglického názvu. Gyrokompas, nazývaný též gyroskop, je přístroj, který je využíván pro určení severu Země v námořní plavbě. Konstrukce gyroskopu se skládá ze dvou do sebe navzájem vložených koulí. Uvnitř se nachází setrvačník. Ten je zavěšený v kardanovém závěsu, tím je zaručena volnost otáčení vzhledem k pohybům celého gyrokompasu. Setrvačník se otáčí rychlostí několika desítek tisíc otáček za minutu. Po dosažení pracovních otáček se osa setrvačníku díky úhlové rychlosti otáčení Země a zemské přitažlivosti ustaví rovnoběžně s místním poledníkem. Tím je nalezen sever. Gyroskop je v lodní dopravě obvykle napojen na autopilota lodi, který z nalezeného severu počítá kurz lodi. Odchylka gyroskopu od pravého severu je maximálně 0,5° při stále rychlosti a kurzu lodi.

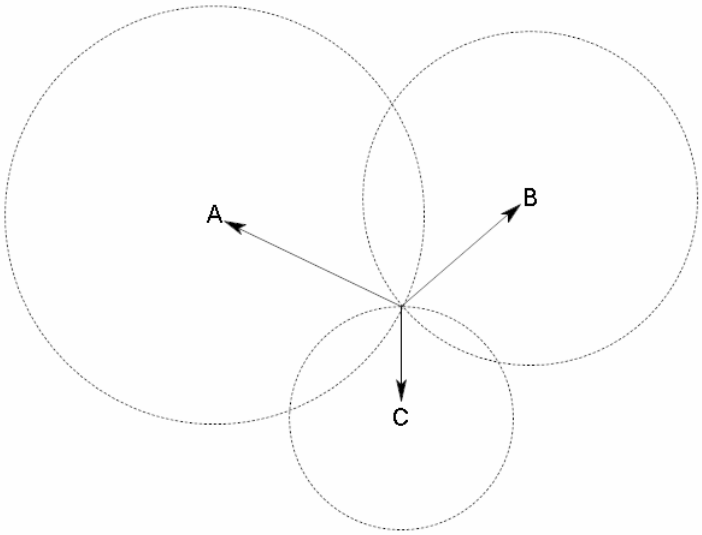
Dříve se využívalo také astronomické určení pozice podle polohy slunce, měsíce a hvězd. Zde byl využíván sextant, což je přenosný přístroj pro měření úhlové vzdálenosti dvou těles nebo úhlu výšky nebeských těles nad horizontem. Sextant využívá překrytí odrazu pozorovaného tělesa v polopropustném zrcadle s obrazem horizontu. Úhel se měří pomocí otočného zrcátka spojeného s kalibrovanou stupnicí, přičemž úhel otočení zrcátka odpovídá výšce nad obzorem. Stupnice mívá

[1] Triangulace - způsob zjišťování souřadnic a vzdáleností v trigonometrii a elementární geometrii.

maximální označenou hodnotu 120°, takže přístroj má tvar kruhové výseče s úhlem 60°. Výše popsané metody určování geografické polohy však nejsou dostatečně přesné a pro účely moderní navigace nevyhovují. V polovině minulého století tedy vědci objevili způsob, jak měřit vzdálenost pomocí radiových signálů. Z tohoto objevu vzešly, do dnešní doby využívané, radiové navigační systémy.

2.1 Radiové navigační systémy

Objev měření vzdálenosti pomocí radiových vln s sebou přinesl vznik pozemních radionavigačních systémů. Pozemní navigační systémy používají radiové signály pro měření vzdálenosti k několika vysílacím věžím umístěným na známé pozici. Radiová navigace pracuje na principu měření vzdáleností k vysílačům. Změří-li přijímač vzdálenost k vysílači A, pak je jasné, že se přijímač nachází na kružnici o poloměru naměřené vzdálenosti kolem vysílače A. Změří-li poté vzdálenost k vysílači B, pak musí být přijímač zároveň na kružnici kolem vysílače A, zároveň na kružnici kolem vysílače B. Nyní se určení polohy zužuje pouze na rozhodnutí, na kterém ze dvou průsečíků kružnic se přijímač nachází. To lze jednoduše zjistit pomocí změření vzdálenosti k třetímu vysílači C.



Obrázek 1. Princip pozemní radiové navigace

Kolem roku 1950 vznikl v USA radiový navigační systém LORAN[2]. Struktura systému LORAN se dle [1] skládá z takzvaných řetězců. Každý LORAN řetězec obsahuje alespoň čtyři vysílače a pokrývá plochu minimálně 500 km^2. Aby se zvýšilo pokrytí tohoto systému, je použito několik řetězců. Celkový rozsah systému LORAN pokrývá celosvětově pouze malou část území, poskytuje pouze dvourozměrné souřadnice a jeho přesnost je dnes přibližně 250 metrů. První globální

[2] LORAN – Long Range Navigation.

navigační systém se stal v roce 1971 americký systém OMEGA. Jeho popis a historie je převzata z [2]. Systém sestával z osmi vysílačů s výkonem 10 kW pracujících na velmi dlouhých vlnách. Vysílače, označeny písmeny A až H, byly vhodně rozmístěny po zemském povrchu. Pro určení rozdílů vzdáleností se používalo měření fázových posuvů. S nástupem satelitních navigačních systémů se stal zastaralým a jeho provoz byl ukončen v roce 1997.

Pro překonání omezení pozemních navigačních systémů byly koncipovány satelitní radionavigační systémy. Vylepšené radiové vysílače byly umístěny na satelity kroužící okolo Země na vysokých drahách. Signály z navigačních satelitů tak mohou pokrýt velkou oblast, několik satelitů může pokrýt celou planetu. Princip činnosti satelitního navigačního systému se podobá sytému pozemní radiové navigace. Zatímco v pozemním navigačním systému jsou vysílače jako orientační body pevně spojeny se zemí a vzdálenosti k nim měřené přijímači, jsou použity pro výpočet dvourozměrné pozice hledáním průsečíku několika kružnic. U satelitních systémů působí satelity jako referenční body a vzdálenosti k nim jsou použity pro výpočet trojrozměrné pozice hledáním průsečíku několika sfér. Poloha vysílačů u pozemního systému je pevně daná, ale satelity obíhají Zemi vysokou rychlostí. Satelity jsou tedy vybaveny zařízením, které v každý okamžik poskytuje přesnou informaci o jejich poloze. Z toho plyne, že přesnost výpočtu geografické polohy závisí i na výpočtu pozice satelitů. Proto je v satelitním systému jejich poloha a dráha průběžně monitorována z několika observačních center, která jsou rozmístěna po celém světě. Tyto observatoře spravuje organizace odpovědná za udržení drah satelitů v odpovídajících hranicích. Podle aktuálních informací o polohách satelitů v předchozích 24 hodinách, jsou předpovězeny jejich dráhy pro dalších 24 hodin. Satelity vysílají informace o svých drahách jako součást struktury svých radiových signálů.

Asi prvním ze satelitních navigačních systému byl systém Transit popsaný v [3]. Jeho hlavním úkolem tehdy bylo určování polohy plavidel. Družice, které systém využíval, byly umístěné na nízkých orbitách, v nadmořské výšce 1 100 km, s dobou oběhu Země asi 106 minut. Ke globálnímu pokrytí bylo třeba souhvězdí pěti družic. Tento systém byl v roce 1964 uvolněn i pro civilní použití a nyní slouží zejména majitelům civilních jachet. Postupem času byl projekt Transit následován řadou dalších systémů.

Například ruským radiovým satelitním systémem Glonass jehož charakteristika je uvedena v [4]. Jeho první tři testovací satelity byly umístěny na orbitu v roce 1982. Původně měl být celý systém spuštěn již v roce 1991, ale nakonec se jeho příprava protáhla až do roku 1995. Glonass používal tři orbitální dráhy s osmi satelity na každé z nich. Celkem tedy bylo na oběžné dráze Země umístěno 24 satelitů, z toho 21 aktivních a tři sloužily jako záložní. Díky rozmístění satelitů byla v každém okamžiku zaručena radiová viditelnost minimálně pěti z nich. Časem však kvůli špatné ekonomické situaci Ruska bylo ponecháno v provozu pouze osm družic. Dnes je jich aktivních dvanáct. Navigační systém Glonass je tedy nyní dostupný pouze v Rusku, severní Evropě a Kanadě, což není špatné vzhledem k tomu, že pracuje pouze 12 satelitů z původních 21. Přesnost tohoto

systému činí v horizontálním měření 57-70 metrů, při vertikálním měření 70 metrů, při měření rychlosti zhruba 15 cm/s a přesnost časování je 1 µs.

Dalším a zároveň nejnovějším družicovým navigačním systémem, je připravovaný evropský systém Galileo. Dle popisu v [5] byl vytvořen a rozplánován v roce 2001. Etapy řešení projetu byly původně naplánovány tak, že vývoj a výzkum bude probíhat do roku 2005, v letech 2006 až 2007 pak proběhne fáze zavádění a od roku 2008 už by měl systém fungovat naplno. Hlavně z finančních důvodů se však jeho zprovoznění zřejmě opozdí. Vzdušnou část Galilea bude tvořit 30 družic, 27 aktivních a 3 záložní. Ty budou létat nad zemským povrchem ve výšce 23 616 km. Jejich trajektorie budou uspořádány do tří oběžných drah, přičemž každá z rovin dráhy bude svírat s rovinou rovníku úhel 56 stupňů. To umožní využívat navigační systém bez potíží až do míst ležících na 75. stupni zeměpisné šířky. Pozemní řídící centra označovaná GCC[3] budou dvě a budou komunikovat s družicemi pomocí patnácti pozemních vysílačů. Vysílání bude pracovat na čtyřech pásmech. E5a se středním kmitočtem 1 176,45 MHz, E5b s kmitočtem 1 207,140 MHz, E6 na kmitočtu 1 278,75 MHz a L1 s kmitočtem 1 575,42 MHz. Galileo bude poskytovat standardní veřejné i komerční signály a speciální signál pro záchranné a bezpečnostní složky. Také bude poskytovat šifrovaný signál pro vládní účely, technicky zabezpečený proti rušení. Podle předpokladů umožní systém Galileo určit aktuální polohu s přesností lepší než jeden metr. Jeho služby by měly být natolik spolehlivé, že na jeho základě bude možné řídit jízdu vlaků, navádět řidiče automobilů a dovést letadla na přistávací dráhu. Sytém Galileo by měl být kompatibilní jak s ruským družicovým navigačním systémem Glonass, tak i s americkým GPS. Galileo vlastně vznikl proto, aby měla Evropa k dispozici vlastní navigační systém, který by nepodléhal libovůli USA, jak je tomu právě u systému GPS. Protože je systém GPS pro řešení tohoto projektu významný, je mu v tomto dokumentu vyhrazena zvláštní kapitola.

2.2 GPS

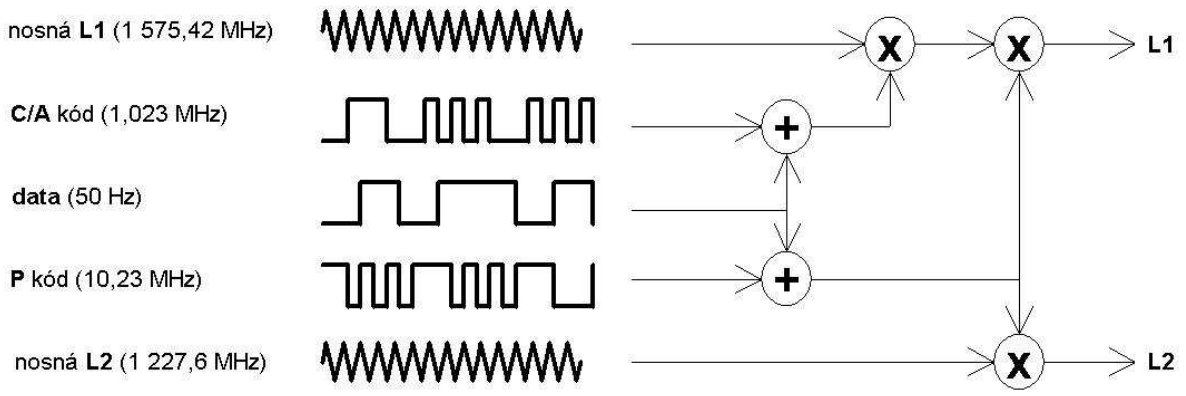
GPS[4] je vojenský navigační družicový systém provozovaný Ministerstvem obrany Spojených států amerických. Jeho charakteristika je uvedena v [6]. Původní název systému, odvozený od jeho funkce, je NAVSTAR[5] GPS. Dnes nesou označení NAVSTAR družice, které GPS využívá ke své činnosti. Vývoj GPS byl zahájen přibližně v roce 1973, k jeho celosvětovému rozšíření došlo v roce 1994, kdy byla na orbitu umístěna kompletní sestava 24 družic. Mezitím se ukázalo mnoho možných využití GPS i v civilním sektoru a proto byl uvolněn i pro běžné použití.

[3] GCC – Galileo Control Center.

[4] GPS – Global Positioning System.

[5] NAVSTAR – Navigation Signal Timing and Ranging Global Positioning System.

Kosmický segment systému GPS tvoří družice o hmotnosti 775 kg, které obíhají na šesti oběžných drahách ve výšce 20 350 km nad Zemí. Jednotlivé oběžné dráhy se od sebe vzájemně odklánějí o 60 stupňů, přičemž na jedné z nich se pohybuje čtyři až pět satelitů. V současnosti je umístěno na orbitě asi 30 satelitů, přičemž aktivních je jich vždy 24, ostatní jsou záložní. Konfigurace družic se na obloze stále mění. Jeden oblet trvá satelitu přibližně 12 hodin, stejné rozmístění se tedy vždy dvakrát denně opakuje. Z jednoho místa na Zemi lze v daný okamžik přijímat signál od 6 až 12 družic. Systém GPS pracuje pouze jednosměrně. Družice pouze signál vysílají a pozemní přijímače pouze signál přijímají. Navigační signál je vysílán v pásmu L (1000 – 2000 MHz). Družice vysílají na několika kmitočtech, které jsou záměrně zvoleny tak, aby byly odolné vůči meteorologickým vlivům. Kmitočet označený L1 (1575,42 MHz), kde je vysílán C/A kód, je dostupný pro civilní uživatele systému GPS. Kmitočet L2 (1227,60 MHz), na kterém je šířen vojenský šifrovaný P/Y kód, je přístupný pouze pro autorizované uživatele, například vojenské služby USA. Vznik kmitočtů L1 a L2 je znázorněn na obrázku číslo 2. Frekvence L3 (1381,05 MHz) obsahuje signály, které souvisí s další funkcí systému GPS. To je odhalování startů balistických raket, jaderných výbuchů a dalších vysokoenergetických zdrojů infračerveného záření. Pro měření ionosférického zpoždění se používá kmitočet L4 (1841,40 MHz). Průchod signálu ionosférou totiž způsobuje přidání dodatečného zpoždění ke zpoždění způsobenému vzdáleností. To se promítne do chyby polohy. Ionosférické zpoždění lze eliminovat, jestliže měříme zpoždění na dvou kmitočtech. Kmitočet L5 (1176,45 MHz) se plánuje použít jako civilní nouzový signál. Tato frekvence spadá do mezinárodně chráněné oblasti letecké navigace, ve které je malé nebo žádné rušení za všech podmínek.



Obrázek 2. Vznik signálu GPS

Každá z družic vysílá posloupnost binárních dat, která se dělí do rámců. Rámec má velikost 1500 bitů a dále se dělí na 5 polí po 300 bitech. Pole se dále dělí na 10 slov. Každé slovo obsahuje 30 bitů, z toho 24 bitů je informačních, ostatních 6 bitů slouží k zabezpečení přenosu. K tomu je použit Hammingův kód[6] se vzdáleností 4. Protože jeden bit trvá 20 ms, je slovo dlouhé 0,6 s, pole

[6] Hammingův kód – binární samoopravný kód.

6 s a rámec 30 s. Obsah polí je následující. První, druhé a třetí pole obsahuje informace o stavu družice. Obsah těchto polí se aktualizuje několikrát za den. Mezi okamžiky aktualizace je jejich obsah konstantní. Čtvrté a páté pole obsahují informace o celém systému GPS. Jejich obsah se aktualizuje několikrát za týden. Mezi jednotlivými aktualizacemi se obsah polí pravidelně opakuje s periodou 25 rámců. Posloupnost 125 polí, ve které je obsažena celá informace o systému GPS, se nazývá navigační zpráva. Pole daného čísla (1 až 5) má 25 možných významů, kterým se říká stránka a označují se pořadovým číslem rámce ve zprávě. První stránka od začátku zprávy má číslo 1. Každý satelit posílá také zprávu o své poloze vyjádřenou tzv. efemeridou, 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. Satelity vysílají také tzv. almanac, což je databáze dalších družic. Tuto databázi si přijímač GPS uloží do paměti ihned po přihlášení a dále si ji aktualizuje. V databázi jsou uloženy kódy okolních satelitů a i jejich přibližná poloha, z níž si přijímač umí odhadnout, kdy se objeví na horizontu. Několik nejbližších kódů si pak přijímač ponechá jako aktuální a každý přijatý signál GPS s nimi porovnává.

pole č.	**obsah**
1	korekce hodin satelitu
2	efemeridy (I)
3	efemeridy (II)
4	další data (IONO, UTC, …)
5	údaje alamanacu pro všechny satelity

Tabulka 1. Rámec signálu GPS

Přenos bitového toku je modulován pomocí modulace BPSK[7]. Podle hodnoty bitu se fáze nosné vlny mění o 180 stupňů. Protože všechny družice vysílají na stejné frekvenci, je nutné jejich signály nějak oddělit. K tomu je využívána metoda CDMA[8], která se nazývá také kódový multiplex. CDMA každý signál družice před vysíláním násobí pseudonáhodnou posloupností s hodnotami +1 nebo -1. Této posloupnosti se říká PN kód. Doba trvání bitu kódu je přibližně 1 μs a je tedy výrazně kratší než doba trvání datového bitu. To způsobí, že vysílaný signál vypadá jako šum. Každá družice používá jiný kód, přičemž tyto kódy se vybírají z množiny posloupností, které jsou vzájemně málo korelované a vedlejší maxima jejich autokorelační funkce[9] jsou zanedbatelná. Autokorelační funkce s malými vedlejšími maximy je vhodná na měření zpoždění signálu, což je základní požadavek pro správné fungování navigačního systému. Slabá vzájemná nekorelovanost je zase základem CDMA. Přijímač provede nejprve před vlastní demodulací BPSK přenásobení přijímaného

[7] BPSK (Binary Phase Shift Keying) – binární klíčování s fázovým posuvem.

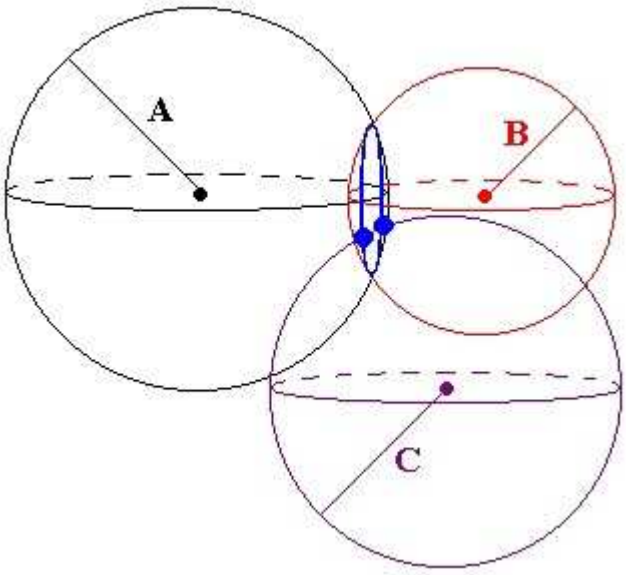
[8] CDMA (Code Division Multiple Access) – kódově rozdělený vícenásobný přístup.

[9] Autokorelační funkce – funkce určující míru podobnosti v rámci téhož signálu.

signálu kódem družice, jejíž signál chceme demodulovat. Signál nechtěné družice se díky nekorelovanosti neobnoví, navíc vypadá jako šum, takže je následnými obvody potlačen.

Řídící a kontrolní segment pozičního systému GPS se skládá z několika monitorovacích stanic. Hlavní řízení je vedeno z ústředí Navstar Headquarters na letecké základně Los Angeles v Californii v USA. Další pozemní stanice se nachází na letecké základně Falcon v Coloradu a hlavní operační řídicí středisko na letecké základně Schriever také v Coloradu. Po světě je rozmístěno dalších pět monitorovacích stanic a 3 povelové stanice.

Princip určení polohy u radiových navigačních systému byl v tomto dokumentu již zmíněn. Přijímač si nejprve vypočte vzdálenost, která jej dělí od několika okolních družic. Tuto vzdálenost vypočte z doby letu signálu a z rychlosti světla včetně započítání vlivů atmosféry. Pokud přijímač zná vzdálenost jen k jedné z družic, předpokládá dle pravidel geometrie, že sám leží někde na plášti koule s poloměrem rovným dané vzdálenosti, jejíž střed tvoří daná družice. Jakmile ale změří vzdálenost i k jinému satelitu, může vypočítat průnik povrchů koule, což je už jen kružnice. Se třetí koulí se možnost polohy zúží pouze na dva body, přičemž jeden z nich leží buď vysoko v prostoru, nebo hluboko v Zemi a může se škrtnou. Tento postup, kterému se říká trilaterace[10], je znázorněn na následujícím obrázku.



Obrázek 3. Princip trilaterace

V praxi je však situace mnohem složitější, protože s měřením a počítáním vzdáleností vznikají nepřesnosti. Z tohoto důvodu se k určení polohy používá vždy nejméně čtyř družic. Chyby mohou vzniknout na úrovni lokalizace satelitů, odchýlením se od skutečné hodnoty rychlosti šíření signálu

[10] Trilaterace – metoda určování relativní polohy objektů pomocí vztahů v trojúhelníku.

atmosférou, vícecestným šířením sígnálu[11], šumem, mezikanálovými odchylkami nebo vlastním měřením GPS přijímače. Jejich vliv na přesnost určení geografické polohy znázorňuje následující tabulka.

zdroj chyby	typické rozmezí
stabilita hodin družice	1 m
lokalizace satelitů	1 m
troposféra	1 m
ionosféra	10 m
pseudonáhodný šum	1 m
šum přijímače	1 m
vícecestné šíření signálu	0,5 m

Tabulka 2. Chyby měření a jejich rozměr

2.3 NMEA

Specifikace NMEA je definována v [7], v této kapitole jsou popsány některé její části, které jsou významné pro řešení toho projektu. NMEA[12] vytvořila specifikaci, která definuje rozhraní mezi jednotlivými částmi námořních elektronických zařízení. Tento standard umožňuje komunikaci těchto zařízení s počítači nebo mezi sebou navzájem. Komunikace přijímačů GPS je definována právě ve specifikaci NMEA. Většina dnešních počítačových programů, které poskytují informace o geografické poloze, očekávají vstupní data v NMEA formátu. Tato data obsahují kompletní informace o pozici, rychlosti a času vypočítané pomocí GPS přijímače. Ideou protokolu NMEA je komunikace na základě posloupnosti dat, zvaných věta. Ty jsou naprosto celistvé a nejsou závislé na dalším datovém toku. Všechny standardní datové věty mají dvoupísmenný prefix. Ten určuje zařízení, které tento typ věty používá. Pro GPS přijímače je tento prefix GP. Prefix je následován třípísmennou sekvencí, jež určuje obsah datové věty. NMEA navíc dovoluje výrobcům hardwaru a zařízení, vytvořit vlastní standardy datových vět, které budou vyhovovat právě jejich potřebám. Všechny tyto věty musí začínat předponou P a dále obsahovat tři písmena identifikující výrobce. Příkladem lze uvést přenosné GPS přijímače Garmin a Magellan, jejichž věty začínají prefixem PGRM respektive PMGN.

Struktura NMEA věty je však poněkud složitější. Každá datová věta musí začínat znakem $ a zakončena musí být sekvencí CR/LF. Dalším omezením je maximální délka věty 80 znaků. Datové položky jsou uvnitř věty odděleny čárkami a jsou reprezentovány prostým ASCII textem. Data mohou mít proměnné rozlišení, například čas nebo pozice mohou být vyjádřeny číslem

[11] Vícecestné šíření signálu – vysílaný signál se k přijímači šíří více cestami. Je to způsobeno odrazem signálu od zemského povrchu nebo odrazem od různých předmětů. Přijaté přímé a odražené signály jsou relativně fázově posunuty a fázové rozdíly jsou úměrné rozdílům v délce dráhy.

[12] NMEA – National Marine Electronics Association.

s několika číslicemi za desetinnou čárkou. Poslední datovou položkou každé věty je její kontrolní součet. Ten se skládá ze znaku * a dvou hexadecimálních čísel, která reprezentují logickou funkci XOR všech předchozích znaků věty mimo znaků $ a *. Tento kontrolní součet může být aplikací využit pro kontrolu správnosti obsahu věty. Standard NMEA, který je přijímači GPS využíván, není zcela ustálený. Existují drobné specifikační odchylky i několik verzí tohoto standardu. Jsou zařízení, které nabízí možnost uživatelského nastavení struktury datových vět. Jiná zařízení mají naopak svůj výstup fixní a nelze je měnit.

Jak již bylo řečeno, NMEA komunikuje pomocí vět. První slovo každé věty, nazvané typ dat, definuje interpretaci zbytku věty. Každý typ dat má svou vlastní jedinečnou interpretaci a je definovaný v NMEA standardu. Význam tohoto řešení spočívá v tom, že kterékoliv zařízení nebo program, které čte vstupní data může zpracovávat pouze ty datové věty, které ho zajímají a ostatní může ignorovat. Standard totiž neobsahuje žádné příkazy, které by GPS přijímači říkaly, co má dělat nebo jaká data posílat. Přijímač tedy neustále posílá všechna data, která zná nebo vypočítá. Přijímající zařízení si na základě typu zprávy vybere jen ty informace, které potřebuje. Zároveň neexistuje způsob jak vynutit data, která nedorazila nebo došlo při jejich přenosu k chybě. Na konci každé věty je tedy pro možnost ověření jejího obsahu umístěn kontrolní součet a v případě chyby musí zařízení nebo aplikace počkat až bude věta znovu zopakována. NMEA standard popisuje mnoho typů datových vět pro všechny druhy zařízení používaných v námořnictví. Některé z nich jsou využívány specielně pro přijímače GPS. Tyto věty začínají prefixem GP. Nejvíce využívané jsou datové typy GGA, GLL, GSA, GSV, RMC a VTG, jež využívá široké spektrum přijímačů. Nejvýznamnější je věta RMC, která nese minimální doporučenou informaci pro navigaci. Příklad a popis této věty je uveden v následující tabulce.

`$GPRMC,170138.615,A,4912.2525,N,01635.0378,E,0.04,16.43,280705,,*32`

položka č.	formát	příklad	popis
1	hhmmss.sss	170138.615	aktuální čas (UTC)
2	c	A	status věty (A = platná, V = neplatná)
3	ddmm.mmmm	4912.2525	zeměpisná šířka
4	c	N	indikátor sever/jih (N = sever, S = jih)
5	ddmm.mmmm	01635.0378	zeměpisná délka
6	c	E	indikátor východ/západ (E = východ, W = západ)
7	d.d	0.04	vodorovná rychlost v uzlech
8	d.d	16.43	kurz pohybu ve stupních
9	ddmmyy	280705	aktuální datum
10	d.d		magnetická deklinace ve stupních
11	c		indikátor východ/západ (E = východ, W = západ)
12	*xx	32	kontrolní součet

Tabulka 3. Příklad věty RMC

Rozhraní NMEA je kompatibilní s běžným počítačovým sériovým rozhraním protokolu RS-232[13]. Rychlost přenosu může být uzpůsobena aktuálním požadavkům, ale typickou rychlostí je ve většině případů 4 800 b/s. Data jsou osmibitová, bez parity a s jedním stop bitem. Všechny jednotky GPS, které používají NMEA protokol by měly podporovat tuto rychlost. Ta je pro potřeby GPS více než dostatečná. Při rychlosti 4 800 b/s může být za jednu sekundu přeneseno 480 znaků. Vzhledem k maximální délce NMEA věty lze tedy za jednu sekundu přenést téměř šest datových vět. NMEA je navrženo tak, aby běželo jako proces na pozadí s tím, že bude neustále posílat datové věty. V závislosti na potřebách aplikace je zpracováván celý datový tok nebo zachytávány informace v pravidelných intervalech. Protokol zaznamenal za několik posledních let mnoho úprav a revizí. Většina současných GPS přijímačů používá standard nazvaný 0183 verze 2. Fyzické spojení přijímače s počítačem, respektive jiným zařízením, lze realizovat pomocí kabelu nebo bezdrátově. U kabelového spojení je využíváno kromě sériového portu také USB. Datový kabel je vybaven pouze dvěma vodiči, datovým výstupem a zemí. Některá řešení obsahují i třetí vodič pro vstup dat do GPS přijímače.

2.4 Nabídka aplikací

Softwarového vybavení, které za pomoci GPS přijímače umožňuje určení geografické polohy, je široké spektrum. Od nenáročných aplikací, jejichž mapy nabízejí pouze jednoduchou vektorovou grafiku až po programové balíky, které dokáží naplánovat trasu cesty, navigovat po ní a dokonce po Internetu zjistit aktuální dopravní situaci. V případě potřeby dokáží najít alternativu v podobě objížďky. Existují dokonce navigační programy, jež dokáží pracovat s digitalizovanou papírovou mapou, tu je však nutné nejprve složitě zkalibrovat. Ač jsou možnosti dnešního navigačního softwaru obrovské, jeho použitelnost je vždy závislá na jediném faktoru. Tím je dostatečná přesnost mapových podkladů. Například mapové pokrytí Evropy často záleží na kupní síle té jednotlivé země, takže zatímco v Německu se člověk může nechat přesně navigovat po celém území státu od čísla popisného k číslu popisnému s tím, že jeho mapa mu nabízí kompletní výčet bodů zájmu, což jsou benzinové stanice, restaurace, ubytovací zařízení, zdravotnická střediska a podobně, tak na Balkáně člověka může překvapit absence silnic první třídy v mapovém podkladu, odlišnost mapy od reálného terénu a totální absence bodů zájmu. Často tedy výběr navigace není určen dle možností aplikace nebo snadnosti ovládání, nýbrž podle kvality mapového podkladu destinace, kde bude navigační systém provozován. Všechny zde zmíněné navigace využívají pro určení geografické polohy družicový systém GPS.

Zřejmě nejrozšířenější aplikací je navigační systém TomTom popisovaný v [8]. Společnost TomTom produkuje navigační software pro PDA a smartphony. Navigace TomTom určená pro PDA

[13] RS-232 – komunikační rozhraní osobních počítačů a další elektroniky.

existuje ve verzi pro Windows Mobile i ve verzi pro PalmOS. Aplikace umožňuje navigování z výchozího do cílového bodu, případně podle předem vytvořeného itineráře. Výpočet trasy lze ovlivnit výběrem typu cesty, automobilová nejrychlejší, nejkratší, případně cesta bez mýtného, nebo pěší trasa. Výsledkem výpočtu trasy je trajektorie cesty zakreslená v mapě, celková vzdálenost a předpokládaná doba cesty. Během vlastní navigace je uživateli vykreslován dvourozměrný nebo třírozměrný mapový podklad, v němž je symbolicky naznačen aktuální směr. V informačním pruhu jsou dále uvedeny informace o vzdálenosti k nejbližší odbočce nebo křižovatce, aktuální rychlost pohybu, název silnice nebo ulice a předpokládaný čas příjezdu do cíle. Vizuální navigaci doplňuje navigace hlasová. V aplikaci lze rovněž nastavit zobrazování bodů zájmu, uživatel může být s předstihem upozorněn, míjí-li vybraný bod zájmu. TomTom také dokáže najít vybraný bod zájmu v okolí uživatele a dovést ho k němu. Za pomocí mobilního připojení k Internetu je možné získávat aktuální dopravní informace. Navigace je potom schopná vyhnout se uzavírce nebo jiným dopravním komplikacím. Aplikace nabízí mnoho různých rozšíření a nadstaveb, žádná z nich však neumožňuje vysílat aktuální polohu na vzdálený počítač.

Velkým konkurentem aplikace TomTom je navigační systém iGO. Ten je vytvářen pro PDA pouze na platformě Windows Mobile. Jeho výhodou je snadnost instalace, protože se software i mapové podklady dodávají společně přímo na paměťové kartě. Po jejím vložení do PDA proběhne instalace automaticky. Aplikace iGO, jejíž vlastnosti jsou uvedeny v [9], nabízí relativně lepší grafické zpracování. Výpočet trasy lze opět ovlivnit. Při samotné navigaci je stejně jako u aplikace TomTom pohled třírozměrný a mapa je automaticky přibližována nebo oddalována v závislosti na aktuální situaci. Hlasová navigace je k dispozici v několika světových jazycích. Zajímavou a ojedinělou funkcí je upozorňování na překročení maximální povolené rychlosti. Také iGO umožňuje zobrazování a vyhledávání bodů zájmu. Co se týče mapového pokrytí, nabízí navigace iGO nejlepší zpracování států východní Evropy. Ani tento navigační systém neumožňuje monitorování polohy na vzdáleném počítači.

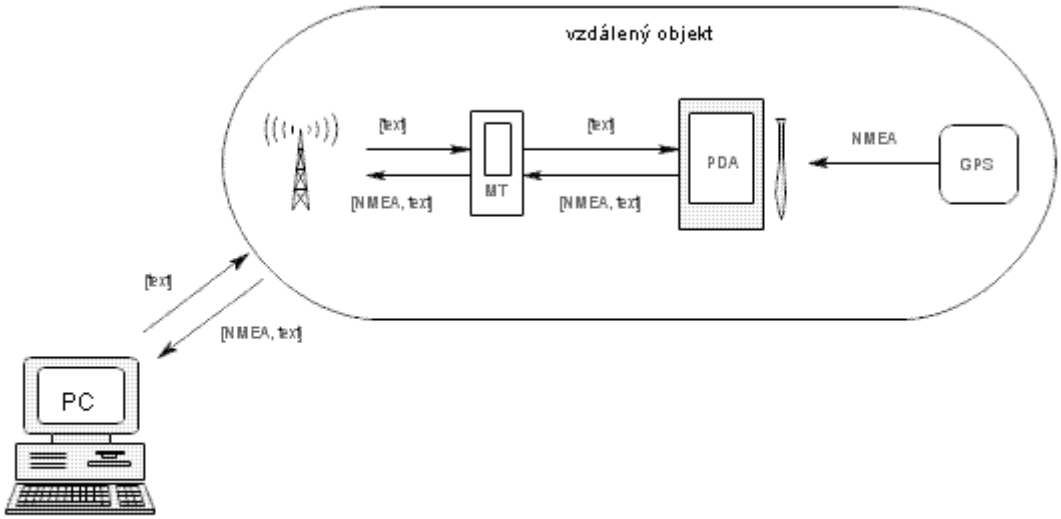
Další významnou aplikací je navigační program Route 66. Vlastnosti aplikace jsou uvedeny v [10]. Route 66 je dosti atypický v tom, že pro jeho běžné používání musí uživatel kombinovat PC a PDA. Vše funguje tak, že se aplikace nainstalujete do počítače, při připojení PDA se nainstaluje navigační program do PDA. Plánování tras probíhá zcela odlišným způsobem, než je tomu u ostatních aplikací. S mapou musí uživatel pracovat nejdříve na PC. Naplánuje trasu včetně průjezdních bodů, poté se teprve mapa exportuje do PDA. Přitom lze nastavit, jak velké okolí trasy a průjezdních, startovních či cílových bodů má být exportováno. Výhodou takového řešení je úspora místa v paměti PDA. Naproti tomu zcela zásadní nevýhodou je nemožnost exportovat jednoduše mapu celé země, exportovat lze pouze obdélníkové výřezy. Navíc když se uživatel například kvůli uzavírce dostanete mimo exportované okolí trasy, nemůže ho již program v PDA navigovat. Vlastní navigační program neumí 3D zobrazení a nezvládá ani zcela plynulý posun mapy. Stejně jako konkurence nabízí hlasovou navigaci. Bohužel ani nabídka bodů zájmu není příliš široká.

Aplikace Microsoft AutoRoute, charakterizovaná v [11], je určená výhradně pro PC. Nabízí mapy celé Evropy a částečně také zbytku světa. Program dále obsahuje poměrně rozsáhlou databázi bodů zájmu. Patří mezi ně některé pamětihodnosti, většina hotelů a množství restaurací. Zároveň je dobře pokryta také železnice s nádražími a stanicemi. Bonusem je položka v menu Phrase book, která obsahuje nejdůležitější výrazy na cestách v řadě evropských jazyků včetně audio nahrávky odpovídající výslovnosti. Stejně jako u konkurenčních produktů je patrný veliký rozdíl v kvalitě map pro západní a východní část Evropy. Plánovaná trasa je v AutoRoute naznačena jak podrobným itinerářem, tak zelenou křivkou na mapě. Záporem programu je velmi slabá spolupráce s GPS modulem. Po připojení přijímače se na mapě zobrazí pouze symbol auta. Chybí jakékoliv udání směru pohybu nebo vzdálenost k dalšímu průjezdnímu bodu. Neexistuje žádná interakce s itinerářem, aplikace ani nenabízí hlasovou navigaci. Výhodou je možnost aktualizovat mapové podklady přes Internet.

Softwarových řešení navigace je celá řada. Žádná další aplikace však není natolik komplexní a použitelná pro navigaci v automobilu. Existují sice řešení, které jsou primárně určeny do automobilu. Jejich možnosti, univerzálnost, mapové pokrytí, možnosti aktualizací a hlavně pořizovací cena je však do jisté míry limituje. Spojení PDA nebo smartphonu a navigačního softwaru TomTom nebo iGO se stalo nejrozšířenějším navigačním systémem v širokém spektru uživatelů. Mapové aplikace Route 66 a Microsoft AutoRoute jsou spíše provozovány na PC a neslouží jako prostředek samotné navigace. Jejich využití v rámci tohoto projektu však má svůj význam. Mohou dobře posloužit na počítači, ze kterého bude sledována poloha vzdáleného objektu. Další variantou, která nyní upevňuje svoji pozici na trhu, jsou jednoúčelová zařízení společnosti TomTom, E-Ten nebo Mio. Ty mohou najít uplatnění kromě automobilu také na motocyklu nebo jízdním kole. Jejich devízou je kromě pořizovací ceny i snadnost ovládání.

3 Analýza a návrh

Družicový navigační systém GPS je v současné době nejdostupnějším prostředkem navigace. Díky relativně nízké ceně přijímačů GPS a také navigačních aplikací, je skupina uživatelů systému GPS poměrně široká. Aplikace, kterou si tento projekt klade za cíl vytvořit, nabízí možnost, která ve výčtu vlastností propracovaných komerčních programů chybí, vzdálené sledování polohy objektu. Smyslem tohoto projektu je tedy navrhnout a implementovat software, který umožní vzdálené sledování polohy aniž by kladl velké nároky na prostředky uživatele. Systém bude využívat pouze těch prostředků, které jsou běžně dostupné a rozšířené. Primární nasazení vytvořené aplikace bude v dopravě. Uživatel, jež se bude pohybovat v dopravním prostředku, bude mít k dispozici PDA, GPS přijímač a mobilní telefon. Na PDA bude moci provozovat libovolný GPS navigační systém. Zároveň bude mít na PDA nainstalovanou a spuštěnou aplikaci, která bude přijímat data NMEA protokolu z přijímače GPS a pomocí mobilního telefonu odesílat tyto data přes Internet na vzdálené PC. Na vzdáleném počítači poběží virtuální ovladač GPS přijímače, který bude data přijímat. S využitím komerčních mapových a navigačních programů a tohoto virtuálního ovladače, bude moci obsluha počítače sledovat vzdálenou polohu uživatele v dopravním prostředku na mapě. Propojení jednotlivých prvků vytvářeného systému schématicky znázorňuje následující obrázek.



Obrázek č. 4. Prostředky systému

Analýza potřeb uživatele vedla ke specifikaci požadavků, kterými se bude řídit návrh aplikace. V prvé řadě byla stanovena platforma, na níž bude systém provozován. Vzhledem k převaze PDA s operačním systémem Windows Mobile padla volba právě na tuto platformu. U osobních počítačů bylo rozhodování jednoduché. Převážná většina mapových počítačových programů existuje pro operační systém Microsoft Windows. S tím souvisí i dalším významný požadavek. Možnost

monitorování vzdálené polohy objektu v libovolném mapovém programu. Protože mapové systémy výlučně komunikují s GPS přijímači připojenými na sériový port, je nutné, aby se vyvíjený projekt tomuto požadavku přizpůsobil. Toho lze docílit dvěma způsoby. První možností je použít null-modem kabel[14]. Pomocí tohoto kabelu je možné například propojit dvě PC a umožnit jejich vzájemnou komunikaci přes sériový port. V tomto projektu by null-modem kabel sloužil k propojení dvou sériových portů jediného PC. Vyvíjená aplikace by odesílala přijaté NMEA věty na jeden z portů, druhý port by byl využit pro příjem dat v mapovém systému. Další možností je použít virtuální null-modem nebo takzvaný virtuální sériový port. To je ovladač[15], který v systému vytvoří dva nové virtuální sériové porty. Tyto porty jsou zájemně propojeny, stejně jako tomu je při propojení fyzickým null-modem kabelem. Programování ovladačů pro operační systém Microsoft Windows patří mezi ty nejnáročnější programátorské disciplíny a vyžaduje to zvládnout poměrně velký rozsah různých navazujících informací. Naštěstí existuje řada komerčních i freeware produktů, které funkci virtuálního null-modemu zastoupí.

V souvislosti s použitím virtuálního ovladače je nutné, aby obě aplikace umožňovaly automatickou detekci dostupných sériových portů na daném zařízení. Vytvořené virtuální sériové porty mohou mít nestandardní název a nelze tedy předpokládat, že by implicitní nabídka sériových portů za každé situace vyhovovala. V opačném případě by byl projekt nepoužitelný, protože by uživatel nemohl zvolit potřebný sériový port.

Na straně vzdáleného objektu, jehož poloha je monitorována, musí být zajištěna komunikace s GPS přijímačem a schopnost odesílat jeho data přes Internet. Aplikace určená pro PDA tedy musí pomocí bluetooth[16] komunikovat s přijímačem GPS a také s mobilním telefonem, který bude sloužit jako prostředník pro komunikaci přes Internet. Velmi důležitým požadavkem je schopnost aplikace na straně PDA ovlivnit objem odesílaných dat a redukovat tak náklady spojené s datovou komunikací mobilního telefonu. Při sledování vzdálené geografické polohy totiž není třeba získávat informaci o pozici v reálném čase, protože v tomto případě není požadována taková přesnost určení pozice objektu. Četnost odesílaných dat z PDA tedy může mít návaznost na nějaký časový interval, případně na vzdálenost ujetou dopravním prostředkem. Díky snížení objemu přenášených dat dojde ke snížení režie za datové spojení, což bude významné zejména při využívání aplikace v zahraničí, kde jsou poplatky za připojení k Internetu vysoké. Ke komfortnímu ovládání aplikace přispěje také její snadná obsluha a uživatelsky příjemný vzhled. Posledním z požadavků je umožnit vzájemnou textovou

[14] Null-modem – kabel standardu RS-232 jehož vodiče pro příjem a vysílání dat a některé další řídící vodiče jsou překříženy.

[15] Ovladač – program, který umožňuje operačnímu systému pracovat se zařízením. Stará se o řízení hardware a zároveň spolupracuje se zbytkem operačního systému pomocí obecnějších rozhraní.

[16] Bluetooth – bezdrátová komunikační technologie sloužící k propojení mezi dvěma nebo více elektronickými zařízeními.

komunikaci mezi sledovaným objektem a obsluhou vzdáleného PC. Díky tomu bude moci obsluha například operativně měnit trasu nebo cíl cesty.

Vytvořený projekt se bude skládat ze dvou aplikací. Jedna z nich je určena pro platformu Microsoft Windows. Problém kompatibility této aplikace napříč různými verzemi tohoto operačního systému není nutné řešit. Prostředky, jež bude program využívat, jsou obsaženy ve všech jeho současně používaných mutacích. U programu, který je určen pro platformu Windows Mobile, je však nutné tento problém zmínit. Windows Mobile je operační systém spojený se základními aplikacemi pro mobilní zařízení jako jsou kapesní počítače, smartphony a zařízení portable media center. Systém je založen na Win32 API z Microsoft Windows, podobá se mu částečně i vzhledem. Platforma Windows Mobile vzešla z projektu Windows CE, který byl realizován v 90. letech. Windows CE byl vytvořen ve verzích 1, 2, 3, 4 a 5. V současné době jsou mezi uživateli stále rozšířeny kapesní počítače s operačním systémem Pocket PC 2002, který spadá do rodiny Windows CE 3.0. Následující generace Windows CE 4.0 s sebou přinesla dnes nejrozšířenější operační systém Pocket PC 2003. Nejnovější PDA však již dnes používají systém Windows Mobile 5.0, ten vychází z nejnovější platformy Windows CE 5.0. Všechny tři verze systému Pocket PC 2002, Pocket PC 2003 a Windows Mobile 5.0 nabízí podporu bluetooth komunikace. Vytvářený projekt je určen pro provoz právě na těchto operačních systémech. Kompatibilita se staršími verzemi Windows CE nemůže být zaručena.

Před samotným návrhem aplikace je nutné přesně specifikovat data, s nimiž bude program pracovat. Na straně sledovaného objektu budou vstupními daty informace získané z GPS přijímače, datové věty NMEA protokolu. Ty jsou složeny, jak již bylo uvedeno dříve, z písmen a znaků ASCII kódu. Při návrhu datové struktury, která ponese tento typ informací, je třeba vzít v úvahu i možnost textové komunikace mezi pohybujícím se objektem a vzdáleným počítačem. Vzhledem k tomu, že obě přenášené informace sestávají pouze ze znaků abecedy a číslic, postačí jako datový typ pouhé pole znaků. Délka tohoto pole bude vycházet z maximální délky datové věty NMEA protokolu. Tato velikost vyhovuje i pro posílání krátkých textových zpráv mezi oběma počítači. Přijímací strana, tedy počítač monitorující polohu objektu, data pouze přijme a předá mapovému programu, který čte data ze sériového portu. Aplikace zároveň musí umožnit odesílat tuto datovou strukturu zpět ke vzdálenému objektu, aby byla zaručena zpětná vazba v textové komunikaci.

3.1 Formální návrh

Návrh aplikace určené pro PDA lze rozdělit do několika částí. V prvé řadě je nutné vytvořit třídu implementující jednoduchý dialog, který bude zajišťovat interakci mezi uživatelem a programem. Tento dialog bude ustavovat spojení s GPS přijímačem i s mobilním telefonem. Zároveň bude obsahovat komponentu textového pole pro odesílání krátkých zpráv na vzdálené PC a komponentu pro vizualizaci aktuální polohy uživatele. Poslední nezbytnou součástí tohoto formuláře bude možnost nastavit objem datového toku a redukovat tak režii za připojení k Internetu. Dále bude

implementována třída pro komunikaci prostřednictvím protokolu bluetooth, definovaném v [12], s přijímačem GPS. Při implementaci této třídy bude využito standardních prostředků, které nabízí vývojové prostředí. Jedním z nich je Bluetooth Manager, který je součástí aplikačního vybavení Windows Mobile. Ten umožňuje výběr a připojení požadovaného zařízení k PDA. Dalším objektem na straně PDA je TCP/IP klient. Architektura TCP/IP[17] je dle [13] členěna do čtyř vrstev. Jsou to aplikační, transportní, síťová a vrstva sítového rozhraní. Toto spojení představuje celou soustavu protokolů, přičemž TCP a IP jsou mezi nimi nejznámější. Protokol TCP vytváří virtuální okruh mezi koncovými aplikacemi, na kterém zaručuje spolehlivý bezztrátový přenos dat ve správném pořadí, plně duplexní spojení a rozlišování aplikací podle portů. Transportní vrstva je realizována právě tímto protokolem. IP je protokol, jež provádí vysílání dat zabalených ve speciální struktuře na základě síťových IP adres obsažených v jejich hlavičce. Vyšším vrstvám poskytuje síťovou službu bez spojení. Program bude tedy s virtuálním ovladačem na vzdáleném počítači spojen pomocí TCP/IP protokolu. Objekt TCP/IP klienta musí zajistit ustavení spojení a realizovat přenos dat mezi oběma aplikacemi.

Návrh aplikace, která je nainstalována na vzdáleném monitorujícím PC, lze opět rozdělit do několika fází. Cílem je navrhnout a posléze vytvořit program, který se z pohledu nadřazené aplikace bude chovat jako ovladač přijímače GPS, jež je k počítači připojen přes sériový port. Nadřazenou aplikací je zde myšlen mapový software, který bude využíván pro sledování vzdáleného objektu. Aplikace bude informace o poloze vzdáleného objektu získávat pomocí TCP/IP spojení z Internetu. Je tedy nezbytné vytvořit třídu TCP/IP serveru. Tento server musí realizovat službu spojení. To zahrnuje navázání spojení, spolehlivý přenos dat ve správném pořadí a ukončení spojení. Aby mohla obsluha počítače komunikovat se vzdáleným objektem za pomoci krátkých textových zpráv, bude vytvořen objekt jednoduchého formuláře. Formulář bude uživatele také informovat o stavu aplikace. Zároveň bude tento objekt předávat datové věty NMEA protokolu mapovému systému pomocí sériového rozhraní. Tyto věty získá pomocí metod třídy TCP/IP serveru. Nezbytnou součástí této aplikace je také možnost zápisu přijatých NMEA vět do souboru. Díky tomu bude možné projekt využít i pro generování knihy jízd, tvorbu různých statistik a archivaci pohybu vzdáleného projektu.

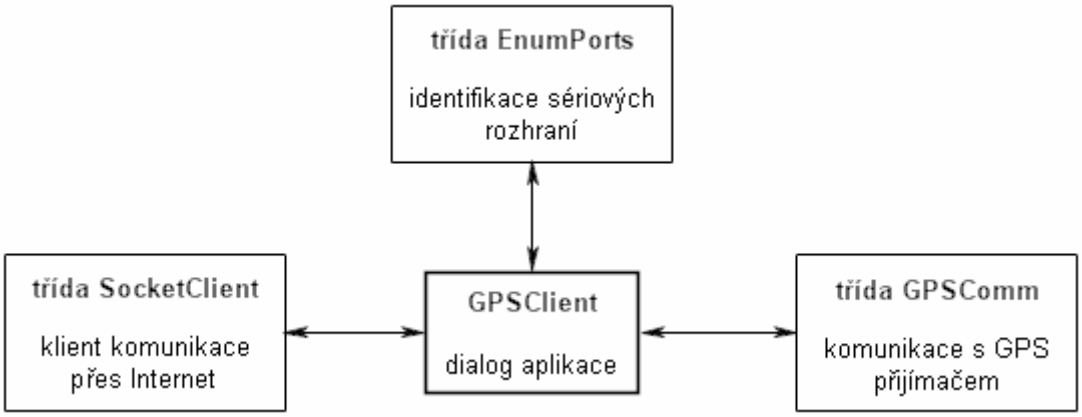
3.2 Objektový model

Na základě předešlého návrhu byl vytvořen objektový model. Paradigmatem objektově orientovaného programování je, podle [14], snaha modelovat při řešení úloh principy reálného světa v počítači pokud možno jedna ku jedné. Metody tohoto programování napodobují vzhled a chování objektu z reálného světa s možností velké abstrakce. Přínosem je větší strukturovanost a modularita vytvářeného programu. Těmito zákonitostmi se řídil také objektový návrh řešeného projektu. Každý

[17] TCP/IP – Transmission Control Protocol/Internet Protocol.

z popsaných objektů bude implementován samostatně a transparentně. Díky tomuto přístupu se výrazně zmenší režie v případě potřeby měnit během implementace návrh, nebo při budoucích úpravách a rozšířeních programu. Využití objektového programovacího jazyka pro implementaci tohoto projektu s sebou přináší nejen možnost budoucího rozšíření o další funkce, ale navíc to umožňuje využití některých objektů k jiným účelům, v jiných projektech než je tento. Modulární stavba aplikace do jisté míry zjednodušuje implementaci. Některé funkce aplikace jsou již připraveny v podobě objektů, které nabízí vývojové prostředí v němž bude projekt vytvářen. Na druhou stranu s sebou tato modulární stavba přináší některé komplikace. V prvé řadě je nutné přesně specifikovat rozhraní objektu. Zapouzdření zaručuje, že objekt nemůže přímo přistupovat k datům jiných objektů, aby nedošlo k nekonzistenci dat. Proto musí každý objekt navenek zpřístupňovat rozhraní, pomocí něhož se s ním pracuje. Rozhraním objektu jsou vlastně metody, které sám obsahuje.

Objektový model aplikace operující na PDA je znázorněn na obrázku č. 5. Obsahuje třídy GPSClient, SocketClient, GPSComm a EnumPorts. Třída GPSClient reprezentuje formulář aplikace. Spravuje jednotlivé komponenty tohoto formuláře, pomocí nichž uživatel komunikuje s programem. Tato třída zároveň importuje objekty tříd SocketClient, GPSComm a EnumPorts. Metody třídy SocketClient jsou využity k navázání spojení se serverem na vzdáleném počítači a odesílání dat zapouzdřených do datagramů[18] přes Internet. Obsahuje také metodu pro ukončení spojení. Metody třídy GPSComm implementují výběr a realizaci spojení mezi PDA a GPS přijímačem. Třída EnumPorts slouží k identifikaci dostupných sériových rozhraní.

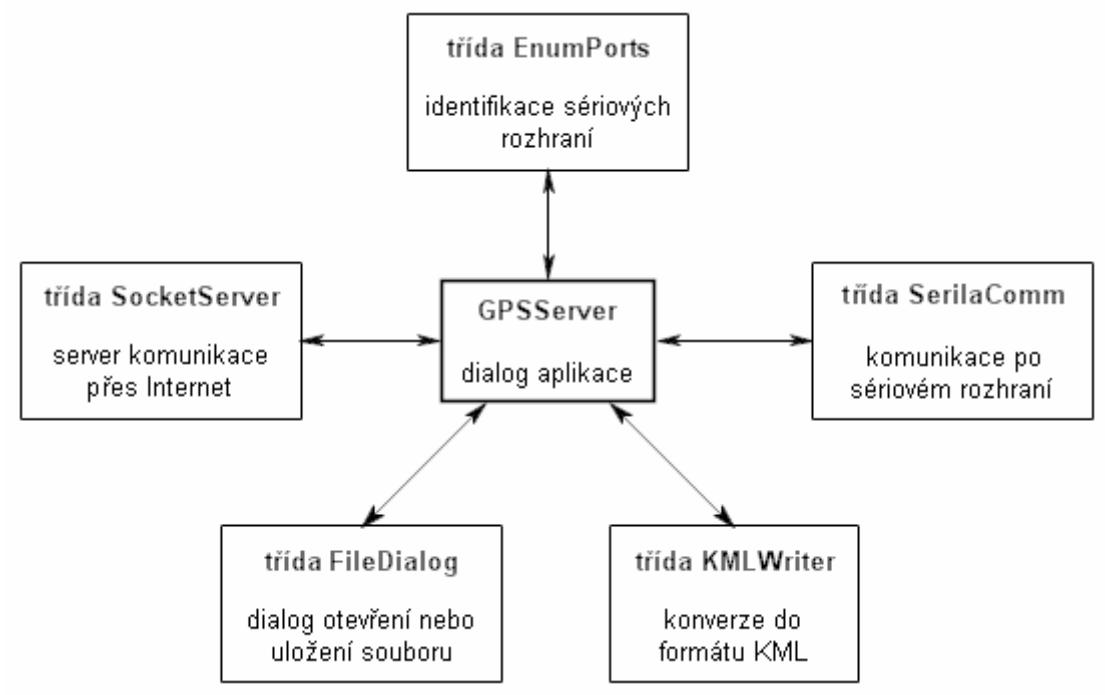


Obrázek č. 5. Objektový model aplikace pro PDA

Model aplikace určené pro osobní počítač obsahuje třídy GPSServer, SocketServer, SerialComm, EnumPorts a FileDialog. Informace z GPS přijímače vzdáleného objektu jsou získávány pomocí metod třídy SocketServer, která implementuje TCP/IP komunikaci se vzdáleným objektem. Stejně jako u objektu třídy SocketClient i třída SocketServer bude obsahovat všechny metody potřebné k navázání spojení s klientem, spolehlivý přenos dat a uzavření komunikace. Hlavní třída

[18] Datagram – datový paket protokolu IP.

GPSServer implementuje dialog aplikace, jeho komponenty a obsluhu těchto komponent. Metody umožňující odesílání NMEA vět přes sériové rozhraní mapové aplikaci budou obsaženy ve třídě SerialComm. Objektový model je znázorněn na následujícím obrázku. Podobně jako u aplikace GPSClient bude uživatel volit pouze z dostupných sériových rozhraní systému. To bude zajištěno implementací třídy EnumPorts. Zvolí-li uživatel možnost archivace NMEA dat do souboru, bude zobrazen dialog pro výběr názvu a místa uložení souboru. Tento dialog bude implementován ve třídě FileDialog.



Obrázek č. 6. Objektový model aplikace pro PC

3.3 Návrh uživatelského rozhraní

Aby byl celý projekt dobře a intuitivně použitelný, je nezbytné kvalitně navrhnout uživatelské rozhraní. Při jeho návrhu bylo zohledněno několik zásad tvorby uživatelského rozhraní. Uživatel by měl v každém okamžiku znát stav aplikace. Tento stav by se měl dovídat odpovídajícím způsobem a v odpovídajícím čase. Aplikace by měla zobrazovat informace v logickém a přirozeném pořadí, s uživatelem by měla komunikovat prostředky a jazykem co nejbližším jeho vnímání světa. Protože se uživatel může dopustit chyby, měl by systém umožnit uživateli dostat se z nechtěného nebo chybového stavu. Aplikace by měla zohledňovat platformové konvence systémů, pro které je vyvíjena. Uživatelské rozhraní by mělo nabízet důležité nástroje a nastavení viditelně, tak aby uživatel vždy danou funkci snadno nalezl. Aplikace by měla mít estetický a minimalistický design. Součástí aplikace musí být adekvátní systém nápovědy, který stručně a srozumitelně povede uživatele k řešení problému.

Na základě těchto kritérií bylo navrženo grafické uživatelské rozhraní obou aplikací. Rozhraní s velkým množstvím ovládacích prvků uživatele odradí, protože vyhlíží příliš složitě. Naopak s ovládacími prvky ukrytými v nabídkách vyhlíží chudě a v případě špatně navržené struktury nabídek je obtížné najít požadovanou volbu. Stejně je tomu u informací sdělovaných rozhraním. Pokud je těchto informací příliš a jsou uspořádány nelogicky, mohou působit přinejmenším matoucím dojmem. Volba počtu a rozmístění prvků uživatelského rozhraní aplikace GPSClient byla provedena s ohledem na velikost displeje PDA, které má rozlišení 240x320 bodů. Vzhledem k použití aplikace v motorovém dopravním prostředku bylo vyžadováno, aby uživatelské rozhraní poskytovalo všechny informace pohromadě na jednom místě. Proto bylo zavrhnuto použití záložek nebo více aplikačních oken. Pomyslným rozdělením programového okna na část s nastavením a část s informacemi se docílilo kompaktního vzhledu a přehlednosti aplikace. Systém chybových hlášení pomocí dialogových oken je pro uživatele, jež nemůže věnovat plnou pozornost běhu aplikace, lepší, než systém se stavovým pruhem.

Na uživatelské rozhraní aplikace GPSServer, která bude provozována na stolním PC, nebyly kladeny tak vysoké nároky. Obrazovka stolního PC nabízí více prostoru pro tvorbu uživatelského rozhraní, proto je jeho návrh snazší. Logické členění hlavního okna aplikace zůstalo zachováno. V horní části budou komponenty nastavení komunikace s klientskou aplikací a mapovým systémem. Ve spodní části potom prvek zobrazující příjem dat od klientské aplikace. Díky tomu, že se bude obsah tohoto prvku dynamicky měnit, bude mít uživatel zároveň kontrolu nad stavem a funkcí celého systému.

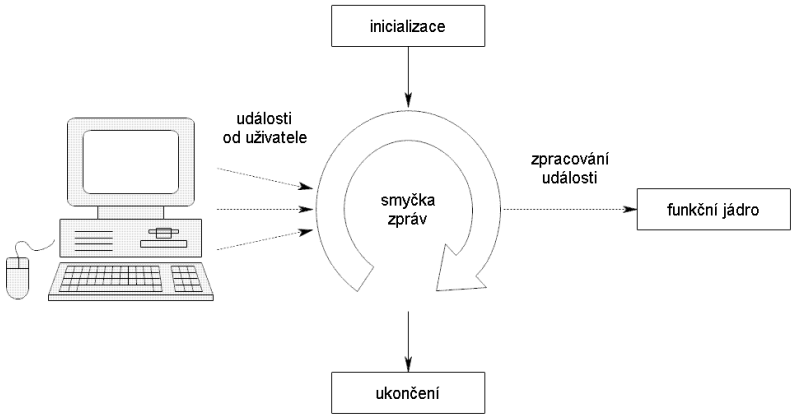
4 Realizace aplikace

Pro vývoj aplikace je důležité zvolit správné nástroje, které tuto práci maximálním způsobem usnadní a urychlí. Nejlepším řešením je vybrat takový nástroj, jenž poskytne veškeré aparáty potřebné od začátku vývoje aplikace až po její dokončení. V případě tohoto projektu byly vybrány dvě vývojová prostředí. Tím prvním je Microsoft eMbedded Visual Tools 3.0. Jedná se o vývojový nástroj pro kapesní počítače, který zahrnuje eMbedded Visual Basic a eMbedded Visual C++ obsahující SDK[19] pro Pocket PC, Palm-Size PC a Handheld PC. Tento nástroj byl využíván při implementaci aplikace pro PDA. Implementace aplikace pro PC byla provedena v prostředí Microsoft Visual Studio 2005. To je balík softwarových produktů, nástrojů a technologií pro rychlou a produktivní tvorbu aplikací využívajících prostředí operačních systémů Microsoft Windows. Obsahuje vývojové prostředí pro jazyky Visual C, Visual C++ a Visual Basic a také .NET jazyky C# a J#. Obě vývojová prostředí nabízí zejména kompletní dokumentaci a nástroje pro dynamickou analýzu kódu za běhu programu, což je pro vývoj takto rozsáhlého a komplikovaného projektu nezbytné.

Implementačním jazykem, jak již bylo zmíněno v návrhu, byl zvolen programovací jazyk C++. Hlavní důvodem, kromě podpory objektově orientovaného programování, byla také mnohem přísnější typová kontrola C++ kompilátoru oproti C kompilátoru. Naproti tomu při vývoji projektu v jazyce C++ není programátor nucen programovat objektově a i nadále může využívat programové moduly, které jsou napsány v jazyce C.

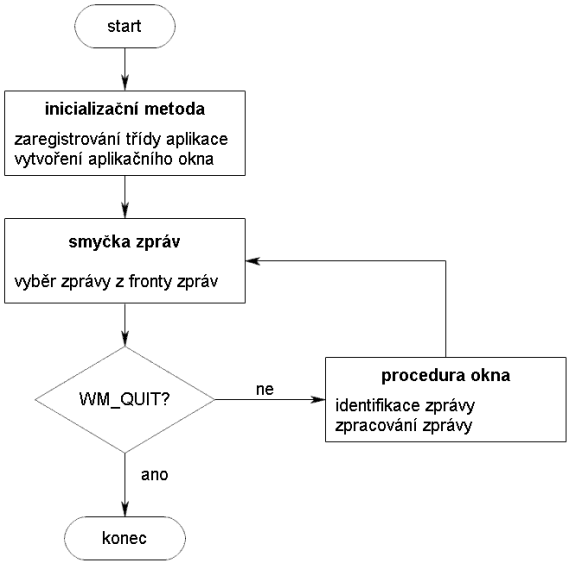
Před popisem vlastní implementace aplikace je nutné si uvědomit, jak jsou vlastně programy s grafickým uživatelským rozhraním ve Windows potažmo Windows CE řízeny a ovládány. V souvislosti s těmito systémy se hovoří o tzv. "událostmi řízeném programování". Jakákoli událost v systému vede k odeslání jedné nebo více zpráv jednotlivým oknům. Tyto zprávy popisují vzniklou událost. Takovou událostí může být pohyb kurzoru, stisk klávesy, uplynutí nějakého aplikací definovaného časovače a mnoho dalších typů zpráv. Zprávy jsou doručovány oknům, jichž se dané události nějak dotýkají. Okna na tyto události mohou na základě přijatých zpráv reagovat. Významným rysem takového systému je schopnost čekat na podněty z nejrůznějších směrů. Popsaný princip je schematicky znázorněn na následujícím obrázku.

[19] SDK – Software Development Kit.



Obrázek č. 7. Událostmi řízené programy

Kostra obou aplikací byla vygenerována automaticky vývojovým prostředím. Základními stavebními kameny každého programu, implementovaného pomocí Windows API[20], je registrace třídy okna a vytvoření okna, smyčka zpráv, procedura okna a zpracování zpráv. Zaregistrování třídy aplikace a procedury okna je provedeno v rámci inicializační metody. Po zaregistrování třídy je možné přejít k samotnému vytvoření aplikačního okna. Smyčka zpráv je cyklus typicky umístěný v hlavní metodě aplikace. V těle cyklu jsou příkazy, které vyjmou zprávu z fronty zpráv aplikace a předají ji k dalšímu zpracování proceduře okna. Výjimkou je zpráva WM_QUIT vedoucí k ukončení cyklu a většinou i celé aplikace. Ve frontě zpráv jsou zprávy náležící všem oknům aplikace, ale v proceduře okna pouze zprávy náležící oknu nebo oknům, patřícím k té třídě, která si tuto proceduru okna zaregistrovala. To znamená, že zprávy náležící oknu této aplikace budou předány právě zaregistrované proceduře okna. Procedura okna identifikuje a zpracovává příchozí zprávy. Na následujícím obrázku je symbolicky popsán běh programu.



Obrázek č. 8. Vývojový diagram aplikace

[20] API – Application Programming Interface.

K vytvoření grafického uživatelského rozhraní aplikací GPSClient a GPSServer bylo využito výhradně funkcí Windows API. API je rozhraní pro programování aplikací. Je to sbírka procedur, metod nebo tříd nějaké knihovny, jež může programátor využívat pro implementaci vytvářeného programu. API definuje vstupní a výstupní parametry funkcí a způsob, jakým se mají funkce knihovny volat ze zdrojového kódu programu. V operačních systémech Microsoft Windows se pro komunikaci programů se systémem využívá rozhraní Windows API. Toto API je navržené pro použití v programovacím jazyce C a C++. Windows API obsahuje několik desítek knihoven. Mimo jiné poskytuje funkce pro tvorbu a obsluhu počítačových oken, tlačítek, posuvníků a dalších ovládacích prvků. Tvorbu standardních dialogových oken pro práci se souborem, volbu fontu nebo barvy a zpracování vstupu z klávesnice nebo polohovacího zařízení. Důvodem použití rozhraní Windows API je efektivnost programového kódu z hlediska minimalizace nároků na operační paměť počítače a další systémové zdroje.

K vytvoření komponent grafického uživatelského rozhraní bylo implementováno několik metod. Konkrétně se jedná metody pro vytvoření textového pole, tlačítka, přepínače, zaškrtávacího políčka a rozbalovacího seznamu. Všechny komponenty jsou vytvořeny pomocí Windows API funkce CreateWindow(). Dále byly vytvořeny metody EnableControl() a SetButtonText(). EnableControl() slouží k aktivaci nebo deaktivaci komponenty. Díky tomu je v projektu docíleno toho, že uživatel může ovládat pouze některé prvky uživatelského rozhraní v závislosti na stavu aplikace. Metoda SetButtonText() mění popisek již vytvořeného tlačítka.

Operační systémy Microsoft Windows na platformách PC i Pocket PC pracují s kódováním znaků Unicode[21]. Při použití Unicode musíme pro znak použít typ WCHAR respektive wchat_t, tyto datové typy jsou 16-ti bitové. Protože obě aplikace umožňují vzájemnou textovou komunikaci, bylo nutné promyslet, jaký datový typ bude pro tuto komunikaci použit. Použití diakritiky v uživatelských zprávách by znamenalo nutnost použít datový typ wchat_t. Tím by se zdvojnásobil datový objem komunikace obou aplikací, což je vzhledem ke snaze datový tok minimalizovat, nepřípustné. Z tohoto důvodu nelze použít v uživatelských zprávách české národní znaky. Programový kód obou aplikací obsahuje velké množství standardních funkcí, které pracují s textovými řetězci. Zejména se jedná o převod z datového typu wchat_t na char a naopak. Pokud to bylo možné, byly použity funkce s bezpečnostním rozšířením. Tyto funkce kontrolují velikost výstupního pole a nedovolí jeho přetečení.

[21] Unicode – kódování používá znaky o velikost 2 byte. Do znakové tabulky se pak vejde 65535 znaků a může obsahovat nejrůznější národní sady znaků.

4.1 Aplikace GPSClient

V první fázi implementace byla vytvořena aplikace GPSClient určená pro PDA. Implementace této aplikace vyžadovala pochopení některých zvláštností systému Pocket PC. Návrh počítal se získáváním informací o aktuální poloze systému z GPS přijímače připojeném k PDA pomocí bluetooth spojení. K bluetooth zařízením je v Pocket PC přistupováno přes sériový port. V Bluetooth Manageru spravující seznam dostupných bluetooth zařízení, je vytvořen takzvaný virtuální sériový port, pomocí něhož je možné k zařízení přistupovat. Z aplikačního hlediska je tedy nutné implementovat sériovou komunikaci. Sériový port, respektive standard RS-232, se používá jako komunikační rozhraní osobních počítačů a další elektroniky. Umožňuje propojení a vzájemnou sériovou komunikaci dvou zařízení. Jednotlivé bity přenášených dat jsou vysílány postupně za sebou po jediném vodiči. Přenos informací probíhá asynchronně, pomocí pevně nastavené přenosové rychlosti a synchronizace sestupnou hranou startovacího impulzu. Při asynchronním přenosu jsou data převáděna do sériového tvaru a vysílána na linku. Synchronizace komunikace je zabezpečena úvodním start-bitem a ukončujícím stop-bitem. Každý vysílaný znak začíná start-bitem, následuje 5-8 datových bitů, případný paritní bit a jeden nebo dva stop-bity. Nejmenší přenášenou informací je jeden znak. Pokud je mezi znaky delší mezera, je vyplněna stop-bitem. Nejvíce používaný formát přenosu je start-bit, 8 datových bitů a stop-bit. Sériové rozhraní také obsahuje mechanismus řízení toku dat. Tento mechanismus popisuje, kdy je možné vyslat znak a kdy ne. Používá se softwarové[22] nebo hardwarové[23] řízení toku dat.

Dalším problémem, na který bylo nutné se zaměřit, byl přenos informací z této aplikace na vzdálený server pomoci TCP/IP spojení. Systémy Pocket PC obsahují takzvaný Connection Manager. Tato utilita slouží k nastavení přístupu PDA do lokální sítě nebo do Internetu. Umožňuje vytvořit profily připojení, které lze aktivovat na základě aktuální potřeby. Díky tomuto přístupu je implementace aplikace komunikující po Internetu do jisté míry zjednodušena. Programátor se totiž nemusí starat o způsob fyzického připojení PDA do sítě, zajímá ho pouze navázaní spojení a vlastní komunikace. Fyzické připojení PDA do sítě je dáno možnostmi konkrétního zařízení a natavením Connection Manageru. Největší výhodou odstínění implementace od způsobu připojení je však univerzálnost vytvořeného kódu. Stejná aplikace může komunikovat přes Internet pomocí GPRS[24] modulu, Wi-Fi[25] karty, kabelového spojení nebo pomocí mobilního telefonu připojeného přes

[22] Softwarové řízení toku dat – určité znaky nebo sekvence přenášených dat jsou rezervovány pro řízení toku dat. Nepoužitelné pro binární přenos dat.

[23] Hardwarové řízení toku dat – k sériovému portu je přidán další vodič. Logická úroveň tohoto vodiče říká, zda je cílové zařízení schopno přijímat či ne.

[24] GPRS – mobilní datová služba přístupná pro uživatele GSM mobilních telefonů.

[25] Wi-Fi – standard pro lokální bezdrátové sítě.

bluetooth, jak je tomu v případě tohoto projektu. Poslední zmíněná možnost však vyžaduje, aby bylo v Bluetooth Manageru správně nastaveno vytáčené spojení pomocí mobilního telefonu. K tomuto účelu je v systémech Pocket PC opět vytvořen virtuální sériový port. Následující text popisuje implementaci a význam jednotlivých tříd aplikace GPSClient. Důraz je kladen zejména na vysvětlení základních principů a uvedení použitých prostředků, nikoli na detaily implementace.

Obrázek č. 9. Aplikace GPSClient

4.1.1 Třída EnumPorts

Programové rozhraní Windows API neposkytuje žádné prostředky, s jejichž pomocí by bylo možné zjistit informace o dostupných komunikačních rozhraní zařízení Pocket PC. Nelze tedy jednoduše zjistit, které sériové porty jsou na daném PDA dostupné. Implementovaná aplikace musí být nezávislá na použitém hardwaru. Nelze tedy implicitně definovat dostupné sériové porty a předpokládat, že toto nastavení bude vždy použitelné. V závislosti na výrobci a modelu zařízení může být pro bluetooth komunikaci využit různý sériový port. Z tohoto důvodu byla implementována právě třída EnumPorts, která najde všechna dostupná sériová rozhraní. Jednou z možností, jak zjistit přítomnost sériových portů, je využít funkce CreateFile(). Tato funkce se standardně používá pro otevření, respektive vytvoření souboru, nebo k inicializaci přístupu k dalším zařízením jako je například sériový port. Funkce vrací handle[26] na soubor nebo zařízení. Postupnou aplikací funkce CreateFile() a testováním její návratové hodnoty by šlo zjistit, které porty jsou v zařízení dostupné a které ne. Toto řešení však

[26] Handle – pomocný objekt bez známé vnitřní struktury. Tento objekt reprezentuje nějaký složitější objekt spravovaný cizím kódem. Vnitřně může být handle reprezentován ukazatelem na cílový objekt nebo strukturou přímo obsahující příslušná data.

není příliš elegantní a v některých případech nemusí fungovat zcela korektně. Při testování virtuálního portu přiřazeného bluetooth komunikaci, by byl navíc zbytečně aktivován Bluetooth Manager, což by bylo pro uživatele matoucí. Druhou možností je zjistit seznam sériových portů přímo z registru[27] systému. Pro přístup k registru poskytuje Windows API dostatek prostředků.

Seznam dostupných komunikačních rozhraní je v Pocket PC uložen v registrech v adresáři HKEY_LOCAL_MACHINE\Drivers\Active. Tento adresář obsahuje složky tedy klíče pojmenované číselnou hodnotou. Tato hodnota může být od 0 do 255, přičemž musí být vyjádřena alespoň dvouciferně. Každý klíč obsahuje položky ClientInfo, Hnd, Key a Name. V konstruktoru třídy je provedena inicializace globálních proměnných. Metoda Enumerate() slouží k hledání dostupných portů a k uložení jejich informací do připravené sruktury. To je provedeno ve dvou fázích. První fáze prochází všechny klíče v adresáři a testuje jejich položku Name. Obsahuje-li položka Name prefix COM, jedná se o sériový port a je zvýšeno počítadlo nalezených portů. Po dokončení první fáze je známo, kolik portů zařízení nabízí a je inicializováno pole patřičné velikosti. Ve druhé fázi jsou klíče v adresáři HKEY_LOCAL_MACHINE\Drivers\Active procházeny stejným způsobem. Název a index nalezených sériových portů je ukládán do připravené struktury. K přístupu k registrům je využito standardních funkcí Windows API. Ty jsou soustředěny v metodě ReadFromRegistry(), která implementuje získávání klíčů registru. Pomocí funkce RegOpenKeyEx() je získán deskriptor typu HKEY ukazující na konkrétní klíč registru. Poté je dvakrát volána funkce RegQueryValueEx(). Při prvním volání je zjištěn datový typ položky klíče, při druhém je získána samotná položka. V případě selhání některého kroku je deskriptor klíče uzavřen funkcí RegCloseKey(). Jakmile jsou získány informace o všech portech, je pole seřazeno vzestupně podle indexu portu. O seřazení se stará privátní metoda SortPorts() využívající algoritmus bubble-sort[28]. Třída také obsahuje metody GetTotalPorts(), GetPortIndex() a GetPortName() pro předání celkového počtu portů, indexu a názvu portu nadřazené třídě. V destruktoru třídy je provedeno uvolnění pole nesoucího informace o nalezených portech.

4.1.2 Třída GPSComm

Třída GPSComm byla vůbec první třídou implementovanou v rámci celého projektu. Předcházelo ji studium principů sériové komunikace a prostředků, které programovací jazyk C++ nabízí. Třída obsahuje několik metod, jež jsou nezbytné pro inicializaci komunikace s GPS přijímačem, čtení přicházejících NMEA vět a zpracovaní některých NMEA vět. Implementace se úzce specializuje na práci s GPS přijímačem. Přestože pracuje se standardním sériovým portem, nemůže být využita pro komunikaci s jiným zařízením, protože čtení dat z GPS přijímače je specifické. Na druhé straně je

[27] Registr – centrální hierarchická databáze, která slouží k ukládání informací potřebných ke konfiguraci systému pro jednoho či více uživatelů, aplikací a hardwarových zařízení.

[28] Bubble-sort – algoritmus, který cyklicky prochází seznam, přičemž porovnává každé dva sousedící prvky. Pokud prvky nejsou ve správném pořadí, prohodí je. Algoritmus běží do té doby, dokud není seznam seřazený.

třída natolik univerzální, že může být využita pro zpracování dat z bezdrátového i kabelového GPS přijímače a to na platformě Pocket PC i PC.

Metoda OpenGPS() slouží k otevření a inicializaci sériového portu. Port je otevřen pomocí standardní funkce CreateFile(). Namísto jména souboru je jako první parametr funkce použit název sériového portu, další parametry jsou shodné jako při vytváření souboru. Další postup inicializace sériové komunikace spočívá v nastavení režimu portu. K tomu je využita struktura DCB[29]. Získání a uložení nového nastavení se realizuje pomocí funkcí GetCommState() a SetCommState(). Pro úplnost jsou ještě nastaveny čekací doby čtecích a zapisovacích operací pomocí funkcí GetCommTimeouts() a SetCommTimeouts(). Metoda CloseGPS() naopak uzavře sériový port. To zároveň vede k ukončení bluetooth komunikace mezi PDA a GPS přijímačem.

Čtení příchozích NMEA vět bylo nejvíce problematickou částí implementace této třídy. Původně bylo čtení závislé na monitorování sériové linky. Pomocí funkce SetCommMask() byly nastaveny události, které chceme na portu sledovat. Následně byla použita blokující funkce WaitCommEvent(), jež předala řízení pokud se na portu objevila požadovaná událost. Pro potřeby tohoto projektu se jednalo o události příchodu nového znaku a aktivace signálů CTS a DSR. Díky, způsobu jakým GPS přijímač vysílá data, však toto řešení nebylo použitelné. Program se nemohl synchronizovat s příchozími daty a čtecí funkce vracela neúplné nebo poškozené NMEA věty. Z tohoto důvodu byla synchronizace pomocí funkce WaitCommEvent() zavrhnuta a její funkci převzala přímo čtecí metoda ReadNMEA(). Tato metoda čeká na příchod znaku $ označující začátek nové věty. Pomocí tohoto znaku se metoda zasynchronizuje a načte zbytek věty do pole. Konec věty je signalizován příchodem znaku *, za nímž následují dva znaky kontrolního součtu a znaky konce řádku CR[30] a LF[31]. Tento způsob může vést ke ztrátě některých vět. V případě, že metoda nestihne zachytit počátek aktuální věty, čeká až na počátek věty další. Vzhledem ke způsobu, jakým GPS přijímače NMEA věty odesílají, je tato ztráta bezvýznamná. Metoda ReadNMEA() tedy pracuje s celými větami předávanými jako pole znaků určité délky.

Třída GPSComm také obsahuje dvě metody pro zpracování NMEA vět. Jedná se o metody ParseGPRMC() a ParseGPGGA(), které zpracovávají stejnojmenné NMEA věty. Volba právě těchto vět je s ohledem na vytvářený projekt pochopitelná. Věta GPRMC je základním určením polohy, rychlosti a času a měla by být podporována všemi GPS přijímači. Věta GPGGA je naopak jediná, která nese informaci o nadmořské výšce. Získané atributy jsou uloženy do příslušné struktury. Struktury pro přenos informací obou vět jsou definovány v hlavičkovém souboru třídy.

[29] DCB – struktura definující řídící nastavení sériové komunikace.

[30] CR (Carriage Return) – netisknutelný řídící znak, který posune kurzor na začátek řádku.

[31] LF (Line Feed) – netisknutelný řídící znak, který posune kurzor na další řádek.

4.1.3 Třída SocketClient

Pro komunikaci s aplikací GPSServer, která bude přijímat NMEA věty a na jejich základě zobrazovat polohu vzdáleného objektu pomocí externí mapové aplikace, byla vytvořena třída SocketClient. Tato třída implementuje navázání a ukončení TCP/IP spojení se serverem, odesílání NMEA vět a příjem uživatelských zpráv od serveru. Pro realizaci teto třídy bylo využito standardního rozhraní Windows Socket API[32]. Rozhodneme-li se v programu používat sockety[33], musíme inicializovat knihovnu socketů pomocí funkce WSAStartup(), jíž předáme ukazatel na instanci struktury WSADATA. Tato funkce nám instanci naplní údaji. Struktura WSADATA slouží k uchovávání informací o implementaci socketů na lokálním počítači. Obsahuje atributy jako číslo verze WSA, popis knihovny socketů, popis stavu socketů, maximální počet použitelných socketů a ukazatel na další informace o konfiguraci počítače. Po ukončení práce se sockety, typicky před ukončením programu, je vhodné zavolat funkci WSACleanup(), jež ukončí používání knihovny winsock.dll. Funkce WSAStartup() a WSACleanup() jsou volány z konstruktoru, respektive destruktoru třídy.

Metoda ConnectToServer() slouží k připojení k serveru. Má dva vstupní parametry, prvním z nich je IP adresa nebo doménové jméno serveru, druhým je číslo portu. Metoda nejprve alokuje TCP/IP socket voláním funkce socket(). Poté je naplněna struktura PHOSTENT pomocí funkce gethostbyname(), která získá odpovídající IP adresu k danému doménovému jménu. Pokud vše proběhne v pořádku, je proveden pokus o navázaní spojení voláním funkce connect(). Selhání některé z výše uvedených funkcí vede k zobrazení chybového hlášení a uzavření vytvořeného socketu. K ukončení spojení byla vytvořena metoda Disconnect() sloužící ke zrušení vytvořeného socketu.

Metody SendToServer() a ReceiveFromServer() implementují odesílání, respektive příjem dat ze serveru. K odesílání je využita standardní funkce send(). Příjem dat je realizován funkcí recv(). Funkce recv() může vrátit chybové hlášení symbolizované konstantou SOCKET_ERROR, nebo počet přijatých znaků. Je-li návratová hodnota funkce 0, pak nebyly přijaty žádné znaky. Vzhledem potřebám tohoto projektu, pracují obě funkce s řetězci znaků. Odesílány jsou NMEA věty a uživatelské zprávy, přijímány pouze uživatelské zprávy.

4.2 Aplikace GPSServer

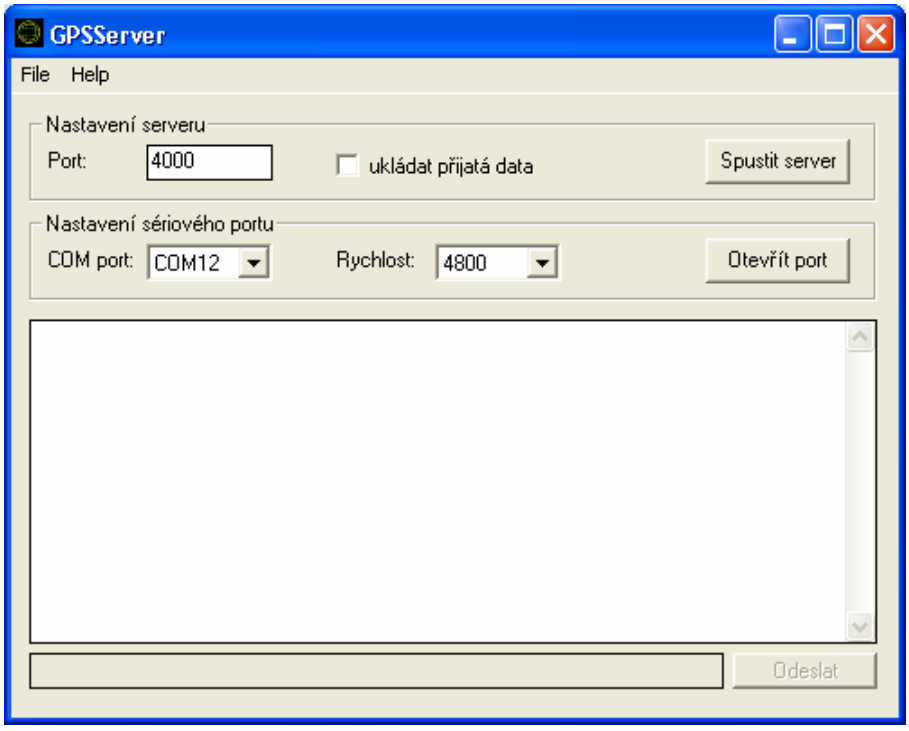
Aplikace GPSServer je určena pro platformu PC. Jejím úkolem je přijímat informace o geografické poloze vzdáleného objektu a s jejich pomocí vizualizovat polohu tohoto objektu. Informace o geografické poloze jsou získávány prostřednictví TCP/IP spojení s klientskou aplikací.

---

[32] Windows Socket API – programové rozhraní, které popisuje jakým způsobem mohou aplikace přistupovat k síťovým službám. Definuje standardní rozhraní mezi aplikací a transportní vrstvou síťového modelu.

[33] Socket – koncový bod komunikace v síti založené na IP protokolu.

Komunikační linkou jsou přenášeny NMEA věty, které aplikace GPSServer musí dále předat prostřednictvím sériového rozhraní mapovému systému. Zároveň musí být zajištěna zpětná vazba mezi serverovou a klientskou aplikací umožňující textovou komunikaci. Formulář aplikace obsahuje komponenty pro ustavení TCP/IP komunikace s klientskou aplikací, pro otevření sériového portu, pro zobrazení přijatých dat a pro odeslání uživatelské zprávy. TCP/IP komunikace je implementována v třídě SocketServer, jež inicializuje server a obsluhuje příjem a odeslání dat. Sériová komunikace je implementována v třídě SerialComm. Problém inicializace sériového rozhraní je velice významný, protože vlastnosti komunikace musí odpovídat reálnému GPS přijímači. Před inicializací sériového rozhraní musí uživatel vybrat ze seznamu dostupných portů. O naplnění tohoto seznamu se stará třída EnumPorts. Ta dokáže z registru systému zjistit všechny dostupné porty. Aplikace také nabízí možnost archivovat přijatá data. Díky tomu je možné trajektorii pohybu vzdáleného objektu později zobrazit nebo použít pro výpočet různých statistik. Uživatel má na výběr ze dvou formátů výstupního souboru. Prvním je soubor s příponou .dat, do něhož jsou zaznamenávány surová NMEA data. Druhou volbou je formát KML[34]. Převod do formátu KML je implementován ve třídě KMLWriter. Zvolí-li uživatel možnost zapisovat přijatá data do souboru, je zobrazen dialog pro uložení souboru. Tento dialog je instancí třídy FileDialog a umožňuje určit místo, název a formát souboru.



Obrázek č. 10. Aplikace GPSServer

[34] KML (Keyhole Markup Language) – je gramatikou jazyka XML a souborovým formátem pro modelování a ukládání geografických dat. Formát KML je využíván zejména aplikacemi Gogole Earth a Gogole Maps.

4.2.1 Třída EnumPorts

Stejně jako v aplikaci GPSClient, bylo pro správnou funkci aplikace nutné identifikovat všechny dostupné sériové porty. Vzhledem k tomu, že aplikace může ke své činnosti využívat virtuální sériový port s nestandardním názvem, je tato potřeba o to víc významnější. Bohužel nebylo možné využít již implementovanou třídu EnumPorts z aplikace GPSClient, protože registr systému Windows CE má jinou strukturu než registr systému Windows pro stolní PC. Především informace o dostupných portech jsou uloženy v jiném umístění a jejich extrakce je odlišná. Z tohoto důvodu by dříve implementovaná třída nefungovala na platformě PC korektně. Informace o používaných sériových portech jsou v registru uloženy v adresáři HKEY_LOCAL_MACHINE\HARDWARE \DEVICEMAP\SERIALCOMM\. Tento adresář obsahuje klíče, jejichž názvem je umístění ovladače portu a hodnotou název portu.

V konstruktoru třídy EnumPorts je provedena inicializace globálních proměnných. Ty nesou informace o počtu dostupných portů, handle klíče registru a strukturu obsahující název portu a umístění ovladače portu. Hlavní metoda třídy, metoda Enumerate(), slouží k identifikaci všech dostupných portů v systému. Metoda nejprve otevře výše zmíněný adresář. Podaří-li se adresář otevřít, je použita funkce RegQueryInfoKey() zjišťující informace o dostupných klíčích v otevřeném adresáři. Z dostupných informací je pro účel metody Enumerate() nejvýznamnější počet dostupných klíčů, který přímo odpovídá počtu dostupných sériových portů. Pokud je tento počet nenulový dojde k inicializaci pole, jež ponese informace o sériových rozhraních v systému. Následně jsou cyklicky získávány názvy klíčů a jejich hodnoty. K tomu je využito standardních API funkcí RegEnumValue() a RegQueryValueEx(). Takto získané informace jsou uloženy do připraveného pole. Pro jednotlivé přístupy do registru systému byly implementovány jednoúčelové privátní metody. Třída také obsahuje veřejné metody pro získání počtu dostupných portů a jednotlivých názvů portů. Jsou to metody GetTotalPorts() respektive GetPortName(). V destruktoru třídy je provedeno uvolnění paměti smazáním alokovaného pole.

4.2.2 Třída FileDialog

Programové rozhraní Windows API obsahuje funkce pro vyvolání standardního systémového dialogu pro otevření nebo uložení souboru. Jsou to funkce GetOpenFileName() a GetSaveFileName(). Parametrem těchto funkcí je ukazatel na strukturu OPENFILENAME. Tato struktura obsahuje vstupní a výstupní parametry obsahující handle rodičovské komponenty, souborový filtr, implicitní příponu souboru, inicializační adresář, název a cestu zvoleného souboru a několik dalších informací. Díky nutnosti inicializovat tuto strukturu by byl programový kód aplikace GPSServer zbytečně nepřehledný, proto byla pro tyto účely implementována zvláštní třída s názvem FileDialog.

V konstruktoru třídy dojde ke zmíněné inicializaci struktury OPENFILENAME. Důležitým krokem je naplnění prvku lStructSize odpovídající velikostí struktury pomocí funkce sizeof(). Dále

dojde k přiřazení globálních proměnných nesoucích název souboru s jeho cestou a samotný název souboru do prvků lpstrFile a lpstrFileTitle. Nezbytné je také inicializovat prvky nMaxFile a nMaxFileTitle, které specifikují velikost polí pro název vybraného souboru. Samotné zobrazení dialogu pro otevření nebo uložení souboru je implementováno v metodách FileOpenDialog() respektive FileSaveDialog(). Třída obsahuje také veřejnou metodu GetFileExtension(). Ta umožňuje jednoduše zjistit, jaký typ souboru pro otevření nebo uložení uživatel zvolil. Rozborem řetězce s názvem souboru je extrahována jeho přípona, jež je vrácena jako návratová hodnota metody. V projektu je této možnosti využito při rozhodování o tom, zda bude aplikace GPSServer ukládat surová NMEA data nebo je bude převádět do formátu KML.

4.2.3 Třída KMLWriter

KML je souborový formát používaný k zobrazení geografických dat v aplikacích Google Earth a Google Maps. V souborech KML se ukládají body, trasy a další informace. KML používá značkovací strukturu založenou na standardu XML. Z pohledu implementované aplikace je formát KML využit pro uložení geografických souřadnic trasy. Díky tomu je možné později znovu zobrazit trajektorii pohybu vzdáleného sledovaného objektu. Souřadnice této trajektorie lze získat rozborem NMEA věty $GPRMC, která obsahuje údaje o zeměpisné šířce a délce.

Základní inicializace je provedena v konstruktoru třídy. Pokud je konstruktor volán bez parametru, je pro uložení použit implicitní soubor track.kml. Ten je vytvořen pomocí privátní metody CreateKMLFile(). Je-li třeba specifikovat jiný soubor, pak je nutné předat handle tohoto souboru konstruktoru. Poslední možností je předat nejen handle souboru, ale také parametry, jež jsou využity při vytváření hlavičky KML souboru. Mezi tyto parametry patří název aplikace, která soubor vytvořila, název trasy, barva a tloušťka trajektorie při zobrazení. Nejsou-li tyto parametry zadány, je použito implicitní nastavení. V konstruktoru je dále volána metoda WriteKMLHeader(). Ta postupně do souboru zapíše všechny potřebné značky až po značku <coordinate>, za níž jsou vkládány souřadnice trasy. Z nadřazené třídy je potom volána funkce WriteCoordinates() ukládající do KML souboru souřadnice. Tyto souřadnice získá rozborem NMEA věty $GPRMC nebo $GPGGA. Souřadnice mají formát X,Y,0, kde X je zeměpisná délka a Y zeměpisná šířka. Jednotlivé souřadnice jsou odděleny znakem mezera. Jsou-li všechny souřadnice zapsány je provedeno zrušení instance třídy KMLWriter. To má za následek vyvolání destruktoru třídy. Destruktor nejdříve pomocí metody WriteKMLFooter() zapíše ukončující párové značky do souboru a poté soubor uzavře.

4.2.4 Třída SerialComm

Komunikace aplikace GPSServer s mapovým programem pomocí sériového rozhraní byla implementována ve třídě SerialComm. Tato třída představuje univerzální prostředek pro komunikaci s libovolným zařízením připojeným na sériový port. Metoda PortInitialize() slouží k otevření sériového portu a nastavení parametrů komunikace. Otevření sériového portu je realizováno voláním standardní funkce CreateFile(). Poté je pomocí funkce GetCommState() získáno aktuální nastavení komunikace a uloženo do struktury DCB. K dosažení co největší univerzálnosti třídy, je většina důležitých nastavení komunikace předána metodě PortInitialize() v parametrech. Díky tomu je možné tuto třídu využít i v jiných projektech bez nutnosti upravovat její programový kód. V rámci tohoto projektu bude pomocí třídy SerialComm simulován reálný GPS přijímač, nastavení komunikace tedy odpovídá jeho požadavkům. Proto je nastavena velikost datového slova na 8 bitů, nastaven jeden stop-bit a vypnuta parita. Po uložení nových parametrů komunikace jsou nastaveny čekací doby čtecích a zapisovacích operací. Jejich hodnoty jsou uloženy ve struktuře COMMTIMEOUTS[35].

Pro čtení a zápis na komunikační linku byly implementovány metody ReadFromPort() a WriteToPort(). Obě metody pracují s řetězci znaků. Ke čtení, respektive zápisu, jsou využity standardní funkce ReadFile(), respektive WriteFile(). Třída také implementuje nastavení monitorovaných událostí a čekání na jejich příchod. Metoda SetPortEvent() nastaví voláním funkce SetCommMask() události, které chceme na portu monitorovat. Událostí lze monitorovat hned několik najednou. Může to být aktivace signálu CTS nebo DSR, chyba na lince, signál RING, příchod nového znaku nebo signalizace odeslání posledního znaku z výstupního pole. Nastavení požadovaných událostí lze provést pomocí funkce SetCommMask() a příslušné konstanty definující událost. Metoda WaitPortEvent() představuje blokující čekání na příchozí událost. Tato metoda předá řízení až v okamžiku objeví-li se na komunikační lince jedna z nastavených událostí. Metoda vrací parametr lpEvtMask obsahující informace o událostech na lince. Pomocí tohoto parametru lze identifikovat, k jaké události na komunikační lince došlo. Poslední metodou třídy je ClosePort(). Tato metoda uzavře handle otevřeného sériového portu voláním funkce CloseHandle() a tím ukončí sériovou komunikaci.

4.2.5 Třída SocketServer

Podobně jako u aplikace GPSClient i zde byla pro TCP/IP komunikaci implementována zvláštní třída. Tato třída tentokrát realizuje funkci TCP/IP serveru. Opět je využito standardního rozhraní Windows Socket API. V konstruktoru třídy je provedeno připojení knihovny Winsock voláním funkce WSAStartup(). Narozdíl od třídy SocketClient implementované pro aplikaci GPSClient, je zde

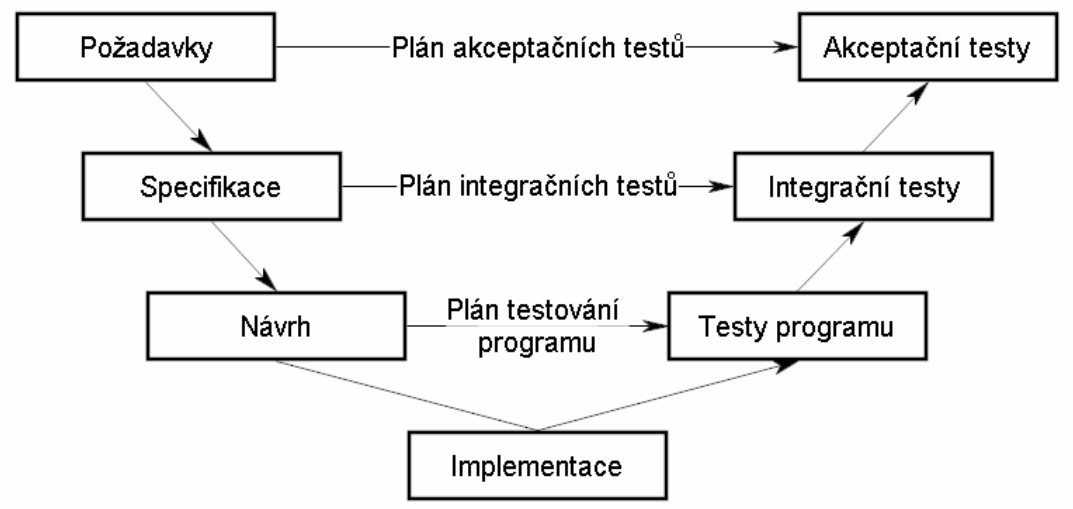
[35] COMMTIMEOUTS – struktura nesoucí informace o čekacích intervalech čtecích a zapisovacích operací při sériové komunikaci.

použito vyšší verze specifikace Windows Socket. Destruktor třídy naopak provede ukončení používání knihovny Winsock pomocí funkce WSACleanup(). Aby bylo možné využít třídu SocketServer i mimo implementovaný projekt, byla navržena a realizována zcela univerzálně. Metoda CreateSocket() slouží k vytvoření socketu a inicializaci komunikace. Vytvoření socketu je realizováno s využitím funkce socket(), které jsou předány parametry AF_INET a SOCK_STREAM. Tyto parametry určují o jakou rodinu protokolů se jedná a že chceme vytvořit TCP/IP spojení. Je-li vytvoření socketu úspěšné, je k němu připojena lokální adresa pomocí funkce bind(). Nakonec je socket převeden do stavu, kdy naslouchá nově příchozím spojením. To je realizováno voláním funkce listen(). Po provedení metody CreateSocket() je socket vytvořen a připraven akceptovat příchozí spojení. Čekání na příchozí spojení ze strany klientské aplikace je implementováno v metodě WaitForConnection(). Metoda používá blokující funkci accept(). Funkce povolí příchozí spojení a inicializuje tak komunikaci mezi klientem a serverem. Funkce accept() je blokující, předá řízení programu až ve chvíli, kdy dojde k navázání spojení nebo k nějaké chybě. Z tohoto důvodu je vhodné volat metodu WaitForConnection() z nového vlákna aplikace. V opačném případě by došlo k zablokování uživatelského rozhraní a další činnosti aplikace. Funkce accept() vrací socket nově navázaného spojení.

Vlastní komunikace mezi klientem a serverem je implementována v metodách ReadConn() a WriteConn(). Metoda ReadConn() využívá funkci recv(). Ta načte příchozí znaky do vstupního pole. Funkce vrací počet načtených znaků nebo SOCKET_ERROR. Odesílání dat je realizováno pomocí funkce send(). Tato funkce odešle obsah výstupního pole. Funkce vrací počet odeslaných znaků. Jestliže návratová hodnota nějaké socketové funkce oznamuje chybu, můžeme zavolat funkci WSAGetLastError() a získat tak kód chyby. Nakonec byla implementována metoda GetLocalAddress() sloužící k získání IP adresy lokálního počítače. Postup získání lokální adresy je následující. Nejprve je zjištěn název počítače voláním funkce gethostname(). Z názvu počítače je pomocí funkce gethostbyname() získána struktura hostent. Struktura hostent slouží k uložení názvu počítače, jeho alternativních jmen a jeho IP adresy. IP adresa lokálního počítače je extrahována ze získané struktury a předána ve formě řetězce jako návratová hodnota metody.

5 Testování

Významnou a také náročnou částí implementace projektu bylo testování vytvořených aplikací. Problematika kontroly tvorby a vyhodnocení kvality software je podstatně složitější problém, než u většiny běžných výrobků a služeb. Proto součástí kvalitního vývoje softwaru jsou dobře zpracované plány testů, které určují jak, co, kdy, kým a čím testovat. Existuje řada metodických přístupů jak testovat funkčnost aplikace během jejího vývoje i po jejím dokončení. Tyto přístupy jsou obvykle poměrně spolehlivé, nicméně časově náročné. Nejlepší cestou vedoucí ke snížení počtu chyb v aplikaci je kvalitní návrh a pečlivá implementace. Proto byl kladen velký důraz na specifikaci požadavků a formální návrh systému. Vývoj projektu se řídil takzvaným V-modelem. Tento model vychází z konceptu potřeby neustálého testování, aby bylo dosaženo vysoké jakost software. Ukazuje nutnost plánovat testy současně se vznikem požadavků na ověřování jednotlivých kroků vývoje software.



Obrázek č. 11. V-model

Již při samotném psaní programového kódu aplikací a použitých tříd, byla testována korektnost použití datových typů. Důraz byl kladen na to, aby nedošlo k překročení krajních hodnot datových typů. Obě aplikace ve velké míře pracují se znakovými řetězci. Proto byly použity bezpečné funkce zabraňující případnému překročení indexu pole nebo přístupu do nealokované paměti. Každá z vytvořených tříd byla testována nezávisle na projektu k externích aplikacích. Tak se prověřila jejich správná funkčnost v různých situacích. Třídy i samotné aplikace mají implementovaný systém chybového hlášení. Selhání kritických kroků vždy vede k zobrazení odpovídajícího chybového hlášení a k výpisu číselného kódu chyby, v nouzovém případě i ke korektnímu ukončení programu. Díky tomu bylo možné během testování snadno identifikovat chybu a vzniklou situaci řešit.

Testováním prošlo také grafické uživatelské rozhranní obou aplikací. Cílem bylo především zjistit, zda některá posloupnost akcí nevede k nepředvídanému chování programu nebo k zablokování uživatelského rozhraní. Jednotlivé komponenty uživatelského rozhraní jsou totiž aktivovány a deaktivovány na základě aktuálního stavu aplikace. Vzhledem k provázanosti stavových proměnných a uživatelského rozhraní však k těmto problémům nemůže dojít. Díky použití vláken pro provádění časově náročných operací jsou ovládací prvky aplikací vždy připraveny reagovat na pokyny uživatele.

5.1 Simulace

Testování systému v reálném provozu je časově náročné, navíc vyžaduje spolupráci více osob. Z důvodu zjednodušení a zrychlení testování byly implementovány dva pomocné programy, které umožnili testovat obě aplikace GPSClient a GPSServer separátně. K aplikaci GPSClient byl vytvořen program pro platformu Pocket PC implementující TCP/IP server, odesílání uživatelských zpráv a ukládání přijatých dat do souboru. Díky tomuto programu bylo možné důkladně otestovat činnost aplikace GPSClient a zároveň získat vzorky dat pro testování aplikace GPSServer. Při testování byly zkoumány zejména reakce aplikace na selhání některých prostředků. Proto bylo úmyslně přerušováno TCP/IP spojení a spojení s GPS přijímačem. Zároveň byla ověřena funkčnost programu při jeho dlouhodobém běhu, protože v praxi bude muset být aplikace v nepřetržitém provozu i několik hodin. Při testování nebyly zjištěny žádné chyby.

K testování aplikace GPSServer byl vytvořen TCP/IP klient, který využíval vzorků dat získaných při testování klientské aplikace. Vzorky dat obsahovaly surová NMEA data, jež byla pořízena za chůze i při vyšších rychlostech v dopravním prostředku. Testovací aplikace četla vzorky dat ze souboru a jednotlivé NMEA věty odesílala aplikaci GPSServer. Tím byla simulována činnost celého systému v provozu. Testování aplikace GPSServer se zaměřilo především na korektní zobrazení polohy sledovaného objektu v libovolném mapovém programu a na konverzi přijatých dat do KML formátu. Také byly simulovány některé kritické stavy, zejména ztráta spojení. Testy serverové aplikace neodhalily žádné nedostatky.

Při simulaci systému i jeho pozdějším testování v reálném provozu bylo pro aplikaci GPSClient využíváno PDA HP iPAQ 5450 s procesorem PXA250 XScale vystavěným na architektuře ARM[36]. Aplikace běžela v prostředí operačního systému Pocket PC 2002.

[36] ARM (Advanced RISC Machine) – 32-bitový RISC procesor, který je určen pro vestavěné systémy.

5.2 Reálný provoz

Jakmile simulace provozu systému potvrdila bezchybnost obou aplikací, byla ověřena funkce systému v reálném provozu. Pro tyto účely bylo vytipováno několik zkušebních tras. Aplikace GPSClient získávala údaje o poloze z bluetooth GPS přijímače a odesílala tato data pomocí mobilního telefonu z jedoucího automobilu. Na vzdáleném PC byla spuštěna aplikace GPSServer, která údaje o poloze předávala mapovému systému. Zároveň byla přijatá data archivována pomocí převodu do KML formátu. Díky tomu bylo možné později zkoumat dosažené výsledky a ověřit správnou funkci celého systému. Pro zobrazení polohy vzdáleného objektu v reálném čase bylo využito mapových aplikací zmíněných v tomto dokumentu. Komunikace mezi aplikací a mapovým programem byla realizována pomocí null-modem kabelu i pomocí virtuálního sériového portu. V reálném provozu se celý systém choval podle očekávání. Drobné odchylky v určení polohy vzdáleného objektu byly způsobeny přesností využitého GPS přijímače nikoli chybou samotného systému.

Následující obrázek zobrazuje jednu z testovacích tras mezi obcí Pardubice a obcí Holice. Testovací trasa byla dlouhá přibližně 22 kilometrů a její zdolání trvalo 35 minut. Aplikace GPSClient byla nastavena tak, aby každých 15 vteřin odesílala na vzdálený počítač NMEA větu GPRMC. Objem přenesených dat mezi oběma aplikacemi činil zhruba 9 kB. Výsledek zápisu trajektorie testovací jízdy byl zobrazen v aplikaci Google Earth.

Obrázek č. 12. Testovací jízda na trase Pardubice - Holice

6 Závěr

Cílem tohoto projektu bylo nastudovat možnosti systémů určujících geografickou polohu a zároveň se seznámit s existujícími programy nebo systémy pro vzdálené sledování polohy objektu. Na základě těchto poznatků navrhnout a implementovat aplikaci pro monitorování vzdáleného objektu. Tento dokument seznamuje čtenáře s pojmem navigace a s prostředky, jež navigace využívá nebo v minulosti využívala. Představuje také některá programová řešení určená pro oblast zjišťování geografické polohy nebo navigace. Nejvýznamnější částí dokumentu je vedení návrhu aplikace a samotná realizace projektu.

Nabídka aplikačních řešení v oblasti navigace je velmi rozsáhlá. Systémů pro monitorování a lokalizaci mobilních objektů je však velmi málo. Navíc jejich zavedení je velmi nákladné a většinou vyžaduje úpravu vozidla nebo jiného sledovaného objektu. Proto vznikla myšlenka vytvořit aplikaci, která monitorování vzdáleného objektu zajistí. Hlavním motivem bylo vytvořit systém s minimálními požadavky a snadnou instalací, tak aby spektrum jeho uživatelů bylo co nejširší. Vzhledem ke skutečnosti, že roste počet uživatelů využívajících navigaci v automobilu, nabízela se možnost vytvořit aplikační vybavení určené pro kapesní počítače PDA, na nichž je navigace často provozována. Díky tomu odpadá investice do dalších zařízení a zavedení celého systému je pouze otázkou instalace aplikace.

Implementace celého projektu byla provedena v programovacím jazyce C++. Projekt byl implementován pomocí standardního rozhraní pro programování aplikací Windows API. Snahou bylo vyhnout se používání prostředků MFC[37]. Aplikace implementovaná výhradně pomocí Windows API je efektivnější a má nižší nároky na operační paměť počítače a další systémové zdroje. Životní cyklus vývoje projektu korespondoval s V-modelem. Tento model přináší do programování aplikace zpětné vazby, které umožňují verifikovat jednotlivé kroky vývoje. Již během vývoje projektu byly jeho jednotlivé části, zejména podpůrné třídy, podrobeny mnoha testům. Díky tomu mohla být většina nedostatků odstraněna již během implementace. Tím bylo zjednodušeno konečné testování celého projektu.

Systém sám o sobě nabízí více možností využití. Primárním cílem projektu je sledování polohy vzdáleného objektu v reálném čase. Díky možnosti archivace přijatých dat aplikace GPSServer lze projekt využít i pro vedení elektronické knihy jízd a různých statistik provozu. Této možnosti bude jistě využito v podnikovém nasazení pro kontrolu využívání služebních vozidel nebo u spedičních společností pro získávání potřebných statistik. Vzhledem k modulární stavbě aplikace a využití výhod objektově orientovaného programování lze program snadno rozšířit o třídu umožňující archivaci trajektorie pohybu vzdáleného objektu do zcela libovolného formátu.

[37] MFC (Microsoft Foundation Class Library) – knihovna zabalující části Windows API do ucelených tříd.

Projekt nabízí celou řadu rozšíření. Velmi zajímavou možností by byla integrace aplikace GPSClient do nějakého nového nebo stávajícího navigačního systému. Pak by bylo možné nejen pasivně sledovat polohu vzdáleného objektu, ale zároveň se na plánování trasy tohoto objektu podílet. Možnost ovlivňovat cíl trasy monitorovaných vozidel by jistě našla uplatnění u spedičních firem nebo v záchranných a bezpečnostních složkách. Pro nasazení systému u větších společností by bylo vhodné pro klientskou aplikaci GPSClient navrhnout vestavěný jednoúčelový systém. Tento systém by integroval GPS přijímač, modul pro komunikaci se serverovou aplikací a další potřebné komponenty. Zároveň by tento vestavěný modul mohl sloužit jako zabezpečení vozidla proti krádeži. Použitím čidel pro detekci vniknutí do vozidla a napojením na jeho řídící jednotku, by systém byl schopen informovat majitele nebo provozovatele vozidla o jeho odcizení a jeho aktuální poloze. Přenos informací by mohl být realizován nejen pomocí sítě GSM, ale také prostřednictvím komunikačních sítí TETRA nebo TETRAPOL[38], které využívají integrované záchranné složky. Výhodou nasazení vestavěného systému je zejména jednoduchost obsluhy takového systému oproti použití PDA a dalších komponent. V případě realizace sériové výroby vestavěného systému by došlo i k snížení nákladů na jeho nasazení do provozu.

Budoucnost družicových navigačních systémů a aplikací s nimi spojených je obrovská. Během relativně krátké doby pronikla původně vojenská technologie do osobních automobilů, do kapesních počítačů a nyní se objevují už takové vymoženosti jako jsou náramkové hodinky s GPS přijímačem. Lidé mají stále větší potřebu identifikovat svoji polohu nebo polohu objektů, o něž se zajímají. Vytvořený projekt má šanci vyplnit prázdné místo v nabídce navigačních aplikací. Tím prázdným místem je oblast sledování polohy vzdáleného objektu.

[38] TETRA, TETRAPOL – digitální radiokomunikační sítě určené pro státní organizace.

Literatura

[1] Loran, Wikipedia – The Free Encyklopedia. Dokument dostupný na URL http://en.wikipedia.org/wiki/LORAN (leden 2008).

[2] OMEGA Navigation system, Wikipedia – The Free Encyklopedia. Dokument dostupný na URL http://en.wikipedia.org/wiki/Omega_Navigation_System (leden 2008).

[3] Transit, Federation of American Scientists. Dokument dostupný na URL http://www.fas.org/spp/military/program/nav/transit.htm (leden 2008).

[4] GLONASS, Wikipedia – The Free Encyklopedia. Dokument dostupný na URL http://en.wikipedia.org/wiki/GLONASS (leden 2008).

[5] Snášel, J. Navigační systém Galileo: o evropské nezávislosti, mobilmania.cz, 2005. Dokument dostupný na URL http://www.mobilmania.cz/default.aspx?article=1111355 (leden 2008).

[6] Dana, P. Global Positioning System Overview, University of Texas, 1994. Dokument dostupný na URL http://www.colorado.edu/geography/gcraft/notes/gps/gps_f.html (leden 2008).

[7] DePriest, D. NMEA data. Dokument dostupný na URL http://www.gpsinformation.org/dale/nmea.htm (leden 2008).

[8] TomTom, portable GPS car navigation system. Dokument dostupný na URL http://www.tomtom.com/index.php (leden 2008).

[9] iGO – GPS navigation software. Dokument dostupný na URL http://www.i-go.com/en/ (leden 2008).

[10] ROUTE 66 Geographic Information Systems. Dokument dostupný na URL http://www.66.com/ (leden 2008).

[11] AutoRoute Home Page. Dokument dostupný na URL http://www.microsoft.com/uk/homepc/autoroute/default.mspx (leden 2008).

[12] Bluetooth, Wikipedie, otevřená encyklopedie. Dokument dostupný na URL http://cs.wikipedia.org/wiki/Bluetooth (leden 2008).

[13] Gilbert, H. Introduction to TCP/IP, 1995. Dokument dostupný na URL http://www.yale.edu/pclt/COMM/TCPIP.HTM (leden 2008).

[14] Objektově orientované programování, Wikipedie, otevřená encyklopedie. Dokument dostupný na URL http://cs.wikipedia.org/wiki/Objektově_orientované_programování (leden 2008).

Seznam příloh

Význam některých NMEA vět

GGA - základní určení polohy, které poskytuje 3D pozici a přesnost údajů.

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

Význam jednotlivých položek:

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 *	

Pokud data neobsahují výšku geoidu, pak může být údaj o nadmořské výšce nekorektní. Některé nestandardní implementace informují o nadmořské výšce s ohledem k elipsoidu spíše než geoidu. Některé jednotky neoznámí záporné nadmořské výšky vůbec. Toto je jediná datová věta, která podává zprávu o nadmořské výšce.

GLL - Věta o zeměpisné šířce a délce, která je pozůstatkem systému Loran. Některé staré jednotky nesmí posílat čas a aktuální informace emulují-li systém Loran. Pokud GPS emuluje systém Loran, lze namísto předpony GP použít předponu LC.

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

Význam jednotlivých položek:

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

GSA - Snížení přesnosti a aktivní družice. Tato věta poskytuje detaily o povaze určení polohy. To zahrnuje počet využívaných družic a údaje o snížení přesnosti měření. Snížení přesnosti je dáno účinkem geometrie drah satelitů. To je vyjádřeno bezjednotkovým číslem, přičemž menší číslo znamená vyšší přesnost. Pro 3D určení polohy je hodnota 1.0 považována za ideální.

`$GPGSA,A,3,04,05,,09,12,,,24,,,,,2.5,1.3,2.1*39`

Význam jednotlivých položek:

GSA	stav satelitů	
A	Automatický výběr 2D nebo 3D určení polohy (M = manuální)	
3	3D určení polohy:	1 = bez určení polohy
		2 = 2D určení polohy
		3 = 3D určení polohy
04,05...	ID satelitů použité pro určení polohy	
2.5	PDOP (snížení přesnosti) v metrech	
1.3	horizontální snížení přesnosti (HDOP) v metrech	
2.1	vertikální snížení přesnosti (VDOP) v metrech	
*39	kontrolní součet	

GSV - Satelity v dohledu. Tato věta poskytuje informace o družicích, jejichž signál je přijímač schopný přijmout. Data jsou čtena z almanachu satelitů. Jedna GSV věta dokáže podat informaci pouze o čtyřech satelitech, k získání plné informace jsou tedy zapotřebí až tři GSV věty. Pole nazvané SNR se v NMEA protokolu často využívá jako indikátor síly signálu. Může být v rozsahu 0 až 99 dB. U některých akčních veličin se však obvykle používají jiné rozsahy, rozdíl bývá zhruba 25 až 35 mezi nejnižší a nejvyšší hodnotou. Hodnotu 0 totiž posílají družice, které jsou sice v dohledu, ale nepodílejí se na určení pozice.

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

Význam jednotlivých položek:

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 – Minimální doporučená informace pro navigaci (Recommended Minimum).

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

Význam jednotlivých položek:

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

VTG - Oprava rychlosti. Při emulaci systému Loran lze opět použít předponu LC.

`$GPVTG,054.7,T,034.4,M,005.5,N,010.2,K*33`

Význam jednotlivých položek:

VTG	oprava rychlosti a dráhy
054.7,T	skutečné dráha ve stupních
034.4,M	magnetická dráha
005.5,N	rychlost v uzlech
010.2,K	rychlost v kilometrech za hodinu
*33	kontrolní součet

Nápověda aplikace GPSClient

GPSClient

Úvodní obrazovka a stručný popis

Ovládání programu je intuitivní. Na obrázku níže je úvodní obrazovka programu.

Obrazovka je rozdělena na dvě poloviny. Vrchní polovina obsahuje ovládací prvky pro nastavení a inicializaci komunikace s GPS přijímačem a serverem. Také obsahuje komponenty pro odeslání uživatelské zprávy. Ve spodní polovině obrazovky jsou zobrazeny aktuální informace o geografické poloze, rychlosti, nadmořské výšce a počtu sledovaných satelitů.

zpět

Připojení GPS přijímače

Z levého seznamu vyberte příslušný sériový port, ke kterému máte připojený GPS přijímač. V pravém seznamu lze specifikovat rychlost komunikace. Předepsaná rychlost pro komunikaci s GPS přijímačem je 4800b/s.

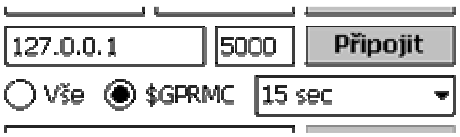
Po stisknutí tlačítka **GPS** dojde k zobrazení okna Bluetooth Manageru, ve kterém je nutné zvolit ikonu zástupce GPS přijímače, případně spárovat GPS přijímač s PDA. Je-li spojení mezi PDA a přijímačem již vytvořené dojde okamžitě k zahájení komunikace. Jakmile se podaří navázat spojení, aktivují se komponenty pro ustavení spojení se vzdáleným počítačem

Zároveň dojde ke změně popisu tlačítka na hodnotu **Ukončit**. Opětovné stisknutí tlačítka vede k ukončení komunikace s GPS přijímačem.

zpět

Připojení k serveru

V levém textovém okně je nutné vyplnit IP adresu nebo doménové jméno vzdáleného počítače. Pravé textové okno slouží ke specifikaci čísla portu. V dalším řádku je nastavení objemu odesílaných dat. V případě zvolení volby **Vše** budou odesílány všechny přijaté NMEA věty. Volba **$GPRMC** zajistí odesílání pouze vět $GPRMC. Při této volbě lze specifikovat interval odesílání dat. Interval lze vybrat z připraveného seznamu. Hodnota 0 sec zajistí, že věty $GPRMC budou odesílány nepřetržitě.



Stisknutí tlačítka **Připojit** vede k navázání TCP/IP spojení se vzdáleným počítačem. Nepodaří-li se spojení navázat, je zobrazeno chybové hlášení, v opačném případě se aktivuje komponenta pro odesílání uživatelských zpráv. Pro odpojení od serveru je nutné stisknout tlačítko **Odpojit**.

zpět

Odeslání uživatelské zprávy

Obsah uživatelské zprávy je nutné napsat do připraveného textového pole. Ve zprávě není možné používat diakritiku.

Stisknutím tlačítka **Odeslat** dojde k odeslání uživatelské zprávy a vymazání textového pole.

zpět

Ukončení aplikace

K ukončení aplikace je třeba stisknout tlačítko **Konec**. Nedojde-li před ukončením aplikace k odpojení od serveru a od GPS přijímače, je zobrazen dialog pro potvrzení.

Stisknutí znaku křížku v pravém horním rohu aplikačního okna nevede k ukončení aplikace, ale pouze k jejímu odeslání do pozadí.

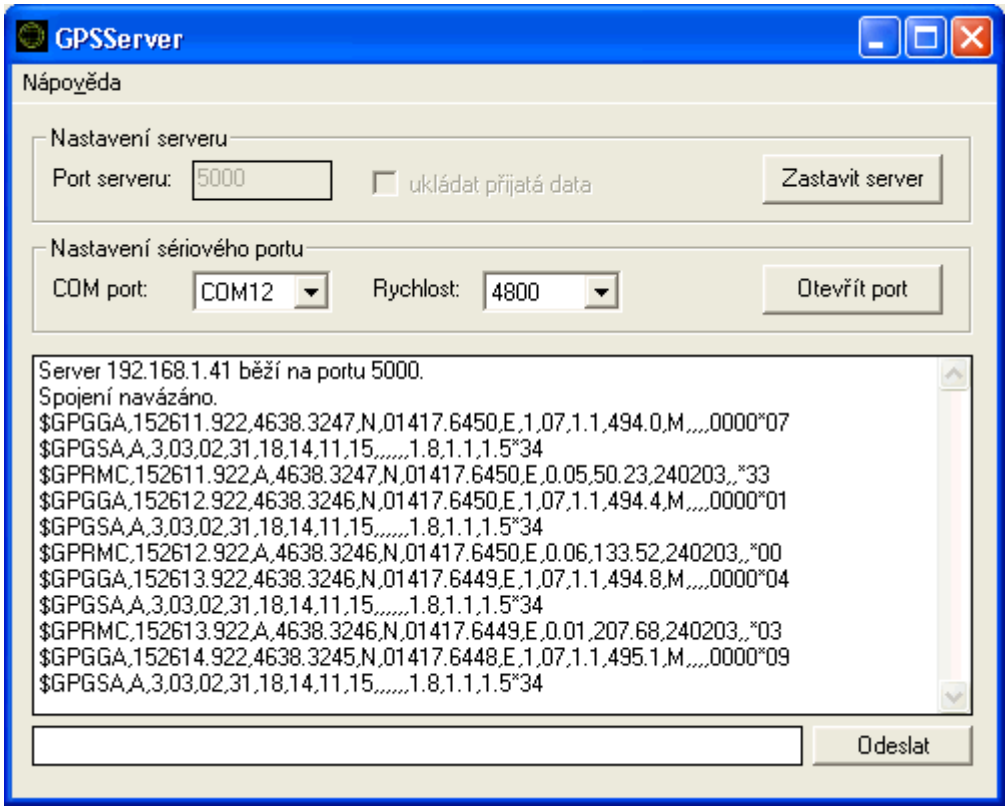
zpět

Nápověda aplikace GPSServer

GPSServer

Úvodní obrazovka a stručný popis

Ovládání programu je intuitivní. Na obrázku níže je úvodní obrazovka programu.

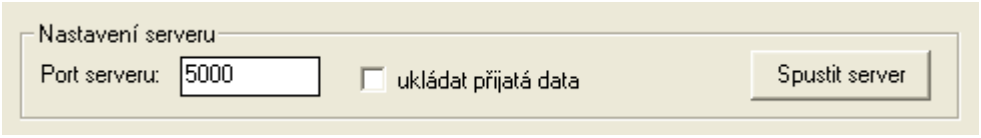


Vrchní část aplikačního okna slouží ke spuštění serveru a nastavení a inicializaci komunikace s mapovou aplikací. Ve spodní části je okno zobrazující stavové informace aplikace a přijatá data. Formulář aplikace také obsahuje textové pole pro zadání a odeslání uživatelské zprávy.

zpět

Spuštění serveru

K ustavení komunikace s klientskou aplikací a k získávání informací o poloze vzdáleného objektu je nutné nejprve spustit server, ke kterému se aplikace GPSClient připojí.



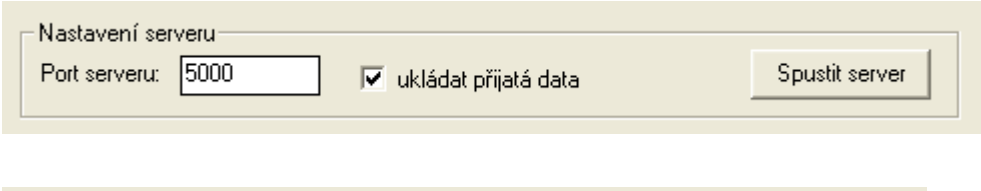
Před spuštěním serveru je nutné specifikovat port, na kterém server poběží. Číslo portu je vhodné volit v rozsahu 5000 - 65535, aby aplikace nekolidovala s jinými službami nebo programy běžícími na počítači. Samotné spuštění serveru se provede stisknutí tlačítka **Spustit server**. O výsledku je člověk informován stavovým hlášením v informačním okně, případně chybovým hlášením.

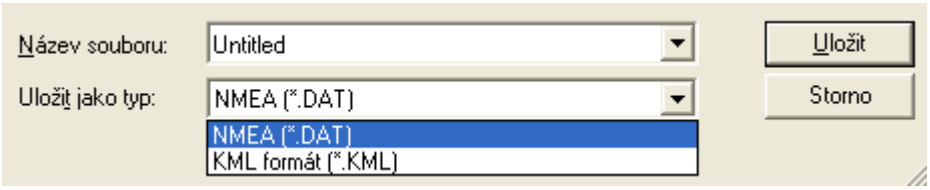
Po spuštění serveru se změní popis tlačítka na **Zastavit server**. Opětovné stisknutí tlačítka vede k ukončení činnosti serveru a k zastavení komunikace s klientskou aplikací.

zpět

Archivace pohybu vzdáleného objektu

Aby bylo možné archivovat pohyb vzdáleného objektu pro budoucí použití, nabízí aplikace GPSServer možnost ukládat přijaté informace o geografické poloze objektu do souboru.



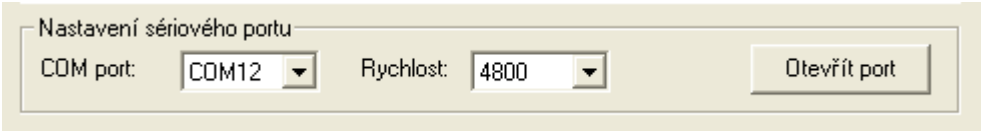


Uživatel má na výběr ze dvou formátů, do kterých se budou přijatá data zapisovat. Implicitní je formát DAT. Při jeho zvolení budou do souboru zapisovány přijaté NMEA věty. Další možností je formát KML. Struktura tohoto formátu vychází z XML. Formát KML lze využít pro zobrazení pohybu vzdáleného objektu v aplikacích Google Earth a Google Maps.

zpět

Otevření sériového portu

Pro zobrazení aktuální polohy vzdáleného objektu v libovolné mapové aplikaci umožňující připojení GPS přijímače je nutné simulovat činnost tohoto přijímače pomocí null-modem kabelu nebo virtuálního ovladače sériového portu.

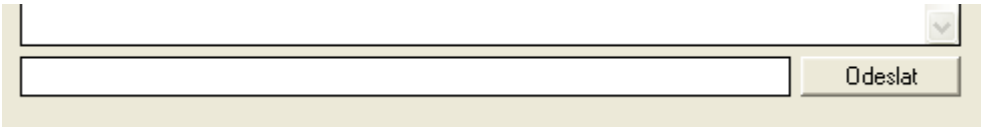


Nejprve je nutné specifikovat sériový port a komunikační rychlost, poté stisknout tlačítko **Otevřít port**. Doporučená komunikační rychlost je 4800b/s. Uzavření sériového portu a ukončení předávání NMEA vět mapové aplikaci se realizuje stisknutím tlačítka **Zavřít port**.

zpět

Odeslání uživatelské zprávy

Obsah uživatelské zprávy je nutné napsat do připraveného textového pole. Ve zprávě není možné používat diakritiku.



Stisknutím tlačítka **Odeslat** dojde k odeslání uživatelské zprávy a vymazání textového pole.

zpět