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

ASISTENT PILOTA BALÓNU PRO IPHONE

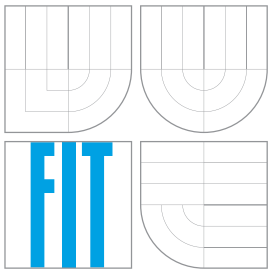
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. ONDŘEJ FABIÁN

BRNO 2011

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

ASISTENT PILOTA BALÓNU PRO IPHONE

BALLOON PILOT ASSISTANT FOR IPHONE

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — Bc. ONDŘEJ FABIÁN
AUTHOR

VEDOUCÍ PRÁCE — Doc Ing. ADAM HEROUT, Ph.D.
SUPERVISOR

BRNO 2011

Abstrakt

Předmětem této práce je vývoj mobilní aplikace pro platformu iPhone, která by sloužila jako asistent pilota horkovzdušného balónu. Hlavní funkčnost spočívá ve vytvoření aktuální mapy směru a síly větru pro jednotlivé letové hladiny a následná navigace na zadaný cíl. Mezi druhotné funkce patří vedení pilotního deníku, záznam a zpětné prohlížení provedených letů. Podstatným aspektem této aplikace je kvalitní návrh uživatelského rozhraní vyhovujícího podmínkám, za kterých bude aplikace použita. Součástí je vyhodnocení experimentu provedeného během reálného balónového letu.

Abstract

This thesis concerns the development of a mobile application for the Apple iPhone platform for hot-air balloon pilots. The main functionality is to create up-to-date map of wind strength and direction for each flight level and navigation tools. Accompanying functions include pilot logbook management, flight recording and recall. A crucial aspect of this application is the GUI design, which meets the conditions in which the application will be used. The application will also be evaluated in test conditions during an actual balloon flight.

Klíčová slova

iPhone, iOS, aplikace, balónové létání, horkovzdušný balón, GPS, navigace, větrná mapa

Keywords

iPhone, iOS, application, ballooning, hot air balloon, GPS, navigation, windgram

Citace

Ondřej Fabián: Asistent pilota balónu pro iPhone, diplomová práce, Brno, FIT VUT v Brně, 2011

Asistent pilota balónu pro iPhone

Prohlášení

Prohlašuji, že jsem tento semestrální projekt vypracoval samostatně pod vedením pana doc Ing. Adama Herouta, Ph.D. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Ondřej Fabián
24. května 2011

Poděkování

Děkuji svému vedoucímu doc Ing. Adamovi Heroutovi, Ph.D. za odborné a užitečné rady při tvorbě práce. Dále děkuji spolužákovi Tomáši Horovi za testování aplikace během balónových letů a vedení společnosti ALFA-HELICOPTER, spol. s r.o. za poskytnutí testovacího zařízení.

Obsah

Seznam obrázků

Kapitola 1

Úvod

Záměrem této práce je vyvinout aplikaci pro mobilní zařízení, která usnadní úlohu pilota horkovzdušného nebo plynového balónu, případně jiného aerostatu. V budoucnosti a za předpokladu dalších rozšíření přesahujících zamýšlený rámec práce by aplikace teoreticky mohla pilota plně zastoupit. Aplikace má za cíl především pomoci pilotovi v rozhodování za účelem dosažení žádaného cíle letu. Mezi ostatní funkce aplikace bude patřit mimo jiné záznam, uchování a zpětné prohlížení provedených letů, vedení pilotního deníku apod. Nejprve bude stručně uvedeno téma balónového létaní. Zdůrazněny budou kritické aspekty této činnosti a informace potřebné pro její vykonávání. V další fázi bude podrobena bližšímu zkoumání zvolená platforma – iPhone – včetně způsobů vývoje a možnosti nasazení. Těžištěm práce bude pak návrh celé aplikace zaměřený zejména na grafické uživatelské rozhraní aplikace. Možnosti uživatelského rozhraní budou ještě před samotnou implementací konzultovány s cílovým uživatelem, pro zajištění co nejsnažšího použití ve specifických podmínkách. Neméně důležitou částí bude také jádro navigace, které bude spočívat ve snímání pohybu balónu v různých letových hladinách pomocí vestavěných senzorů a následně vypočítávat vektor síly větru pro dané hladiny. Během počáteční fáze letu získá tak aplikace jisté povědomí o aktuální meteorologické situaci, které bude využito pro následnou navigaci na zadaný cíl. Závěrem bude vyhodnocen dosažený výsledek včetně experimentu provedeného během reálného balónového letu.

Kapitola 2

Balónové létání

Nejprve budou zmíněny základy a počátky balónového létání. Dále bude přiblížena konstrukce balónu, jeho ovládání a způsoby využití pro komerční, rekreační a soutěžní účely.

2.1 Historie

První mezinárodně uznávaný volný let s pasažéry na palubě se konal 21. listopadu roku 1783 v Paříži. O konstrukci horkovzdušného balónu, který byl k tomuto letu využit, se zasloužili bratři Joseph-Michel a Jasques-Etienne Montgolfier. Podle nich se horkovzdušnému balónu říká také montgolfiéra. Původně měli být pasažéry odsouzení vězni, ale francouský vědec a průkopník létání Jean-François Pilâtre de Rozier přesvědčil krále Ludvíka XVI., aby on a Marquis Francois d'Arlandes mohli být prvními pasažéry. Jejich úspěchu předcházelo několik více či méně úspěšných pokusů s horkovzdušnými i plynovými balóny. Podrobný výčet sestavila CIA (Commission Internationale d'Aerostation) k nalezení na [2].

První horkovzdušný balón moderního typu sestavil v roce 1960 američan Paul Edward Yost. Použil plastový obal a kerosin (letecký petrolej) jako palivo, viz [2].

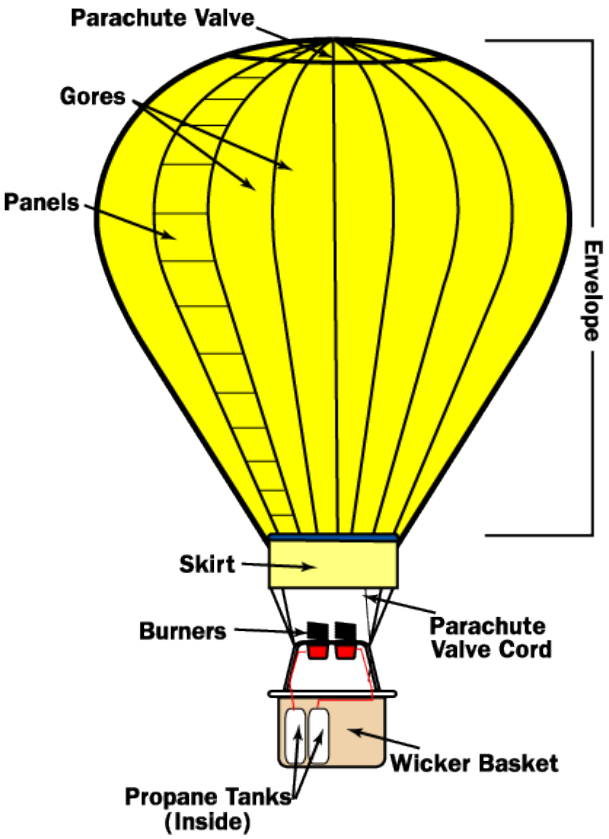
2.2 Konstrukce balónu

Moderní balóny mají dvě základní součásti obal a koš. Obal neboli plášť je vyrobený z odolné polyesterové látky schopné snášet vysoké teploty. Skládá se zpravidla ze 8–24 vertikálně uspořádaných pásů látky nazývaných poledníky. Koš je většinou proutěný a je schopný pojmout u běžných balónu posádku o 3–20 členech. Princip balónového létání spočívá ve využití vztlakové síly. Je-li plášť naplněn plynem, který je lehčí než okolní vzduch, je možné pod něj zavěsit koš. Pokud je balón jako celek také lehčí než vzduch, vznáší se bez pohonné jednotky. Pilot balónu jej může ovládat pouze změnou vztlakové síly plynu uvnitř pláště čili pouze po vertikální ose. Způsob, jakým toho dosáhne, se liší v závislosti na typu balónu. Ostatní pohyb balónu je dán pouze okolním větrem, který se liší v různých letových hladinách. Hlavní myšlenkou pilotování je tedy navádění balónu do takové výšky, ve které fouká vítr směrem k cíli. Tuto hladinu však není jednoduché najít.

V současnosti se odlišují tři typy moderních balónů v závislosti na náplni pláště a způsobu ovládání.

- *Horkovzdušný balón* je, jak napovídá jeho název, naplněn horkým vzduchem, který má při stejném objemu nižší hmotnost než studený vzduch. Tento typ balónu je nejběžněji využívaný pro všechny typy balónového létání. Kromě obalu a koše musí být součástí

také vysokotlaký difuzní hořák a palivové láhve uložené v koši. Jako palivo je nejčastěji použit propan nebo méně účinný a levnější propan butan. Ohřevem vzduchu v plášti dojde ke zvýšení vztlakové síly působící na balón a ke vzestupu. Klesání nastává, pokud pilot nechá samovolně obal ochladit nebo pokud otevře větrací otvor umístěný ve vrcholu pláště. Běžný let trvá přibližně jednu hodinu a pokrývá vzdálenost zhruba 20 kilometrů. Tento fakt vezmeme v úvahu později při návrhu aplikace v souvislostí s výdrží baterie mobilního zařízení. Jelikož tuto dobu vydrží zařízení při plné zátěži bez problémů, nemusí aplikace šetřit energií a může se soustředit na výkon.



Obrázek 2.1: Schéma horkovzdučmého balónu a jeho jednotlivé části. Zdroj [11]

- *Plynový balón* je napuštěn plynem s nižší molární hmotností, než má okolní vzduch. Mezi plyny, které jsou pro tento účel v různé míře využívány, patří vodík, helium, amoniak, svítiplyn a metan. Nejčastější je helium, jelikož je na rozdíl od vodíku nehořlavé a má velice nízkou molární hmotnost. Je však ve srovnání s ostatními plyny poměrně drahé. V koši plynového balónu je obvykle umístěno závaží, které je odhazováno a tím zajištěno stoupání. Klesání dosáhneme odpouštěním nosného plynu.

- *Rozierův balón* je hybridní kombinací dvou výše zmíněných typů s oddělenými komorami pro horký vzduch a nosný plyn. Stačí mu daleko menší množství horkého vzduchu k manévrování a má tak mnohem menší spotřebu paliva. Kvůli těmto vlastnostem se Roziere využívá například k dálkovým letům.

Podle FAI (Fédération Aéronautique Internationale) Ballooning commission - CIA [8] rozlišujeme 15 kategorií horkovzdušných balónů dle velikosti pláště od nejmenší AX-1 (méně

než 250 m^3) až po největší AX-15 (více než 22000 m^3). Kromě velikosti se balóny mohou lišit i tvarem pláště. Pomineme-li reklamní balóny bizardních tvarů, jako pivní plechovka, hrad nebo želva, liší se balóny především v průměru a výšce pláště potažmo v počtu poledníků. Sportovní balóny jsou vyšší a mají menší průměr. Tento tvar jim umožňuje velmi rychle stoupat a klesat. Nejextrémnější balóny jsou schopné vyvinout vertikální rychlost až 9 ms^{-1}.

2.3 Využití balónu

Pro účely této práce můžeme balónové létání rozdělit do dvou skupin. Do první zařaďme ty činnosti, u kterých nezáleží, kde přistaneme, ale cílem je samotný let. Mezi takové disciplíny patří např. rekreační – vyhlídkové – lety, balónová reklama apod. Druhou, zajímavější skupinou, u které záleží na místě přistání nebo průletu, nazveme sportovní létání.

Sportovní létání se člení do mnoha disciplín. Pro představu uveďme několik málo z nich uvedených na [11] dle oficiálních pravidel pro soutěže horkovzdučných balónů:

- *Rozhodčím určený cíl* je disciplína, při které balónové posádky ještě před samotným letem obdrží zadání letu. Součástí zadání je čas startu — tzv. startovní okno, místo startu, vymezený prostor pro disciplínu, čas pro dokončení disciplíny a místa, kde jsou umístěny cíle v podobě křížů na zemi. Tyto cíle se snaží posádka zasáhnout *markery*, což jsou 140 cm dlouhé a 10 cm široké kusy látky opatřené závažím s pískem o hmotnosti 70 g. Každému týmu je před startem přidělen *GPS logger*, který zaznamenává celý průběh letu a je po přistání odevzdán hlavnímu rozhodčímu k vyhodnocení.

- *Váhavý valčík* je velice podobný jako předchozí disciplína. Rozhodčím jsou ale vytyčeny tři cíle, mezi kterými si pilot může vybrat.

- *Let na cíl* je disciplína, při které rozhodčí určí maximalní a minimální vzdálenost od cíle a v tomto rozsahu si pilot volí místo startu.

- *Hon na lišku* a *potopení lodi* probíhá ve snaze odhodit marker co nejblíže místa přistání jiného nesoutěžního balónu – lišky, lodi. Jediný rozdíl v těchto disciplínách je místo startu nesoutěžního balónu. Při honu na lišku startují soutěžící ze stejného místa jako liška.

- *Nejmenší* nebo *největší vzdálenost* spočívá ve snaze urazit v daném čase minimální nebo maximální vzdálenost od místa startu.

Závod horkovzdušných balónů je zpravidla vícedenní akce. Ať už vybereme jakoukoliv disciplínu, vždy je nutné vést balón s maximální přestností na cíl. Pilot toho dosáhne pečlivým výběrem letových hladin a tím požadovaného kursu. Právě pomoc při volbě letové hladiny by měla být primárním cílem výsledné aplikace této práce.

2.4 Ostatní náležitosti létání

Balónové létání jako každá jiná forma létání musí splňovat jisté právní a jiné náležitosti. Oficiálními předpisy L a JAR státního podniku Řízení letového provozu ČR je možné dohledat na webových stránkách Letecké informační služby viz [15]. Mezi nimi lze nalézt také meteorologické a jiné podmínky, za kterých je možné startovat. Na palubě každého balónu

se musí nacházet povinná sada přístrojů: výškoměr, variometr (ukazatel vertikální rychlosti), palivoměr na tlakových lahvích a tavná pojistka indikující maximální přípustnou teplotu vzduchu v obalu.

Způsobilost pilota k létání je podmíněna splněním pilotních zkoušek dle předpisů UCL a získáním kvalifikace HOT pro horkovzdušné balóny. Pro účely výcviku a prodlužování platnosti pilotní kvalifikace si každý pilot vede tzv. *logbook* čili pilotní deník, ve kterém zaznamenává pro každý let místa a časy startu a přistání, posádku, spotřebované palivo a další informace.

Během letu musí pilot komunikovat rádiem s řízením letového provozu pro oblast, kde se právě nachází. Dodržovat pravidla letu za viditelnosti VFR, případně letu podle přístrojů IFR. Respektovat musí také minimální výšku letu, která je v obydlené oblasti 300 m nad zástavbou a mimo obydlenou oblast 150 m nad terénem. Další omezení pro pilota jsou zakázané zóny dané a ohraničením a minimální anebo maximální výškou např. v blízkosti letišť nebo elektráren.

Informace o meteosituaci, vedení pilotního deníku a upozornění na blízkost zakázaných zón jsou také činnosti, které by bylo možné dobře zjednodušit pomocí mobilního zařízení.

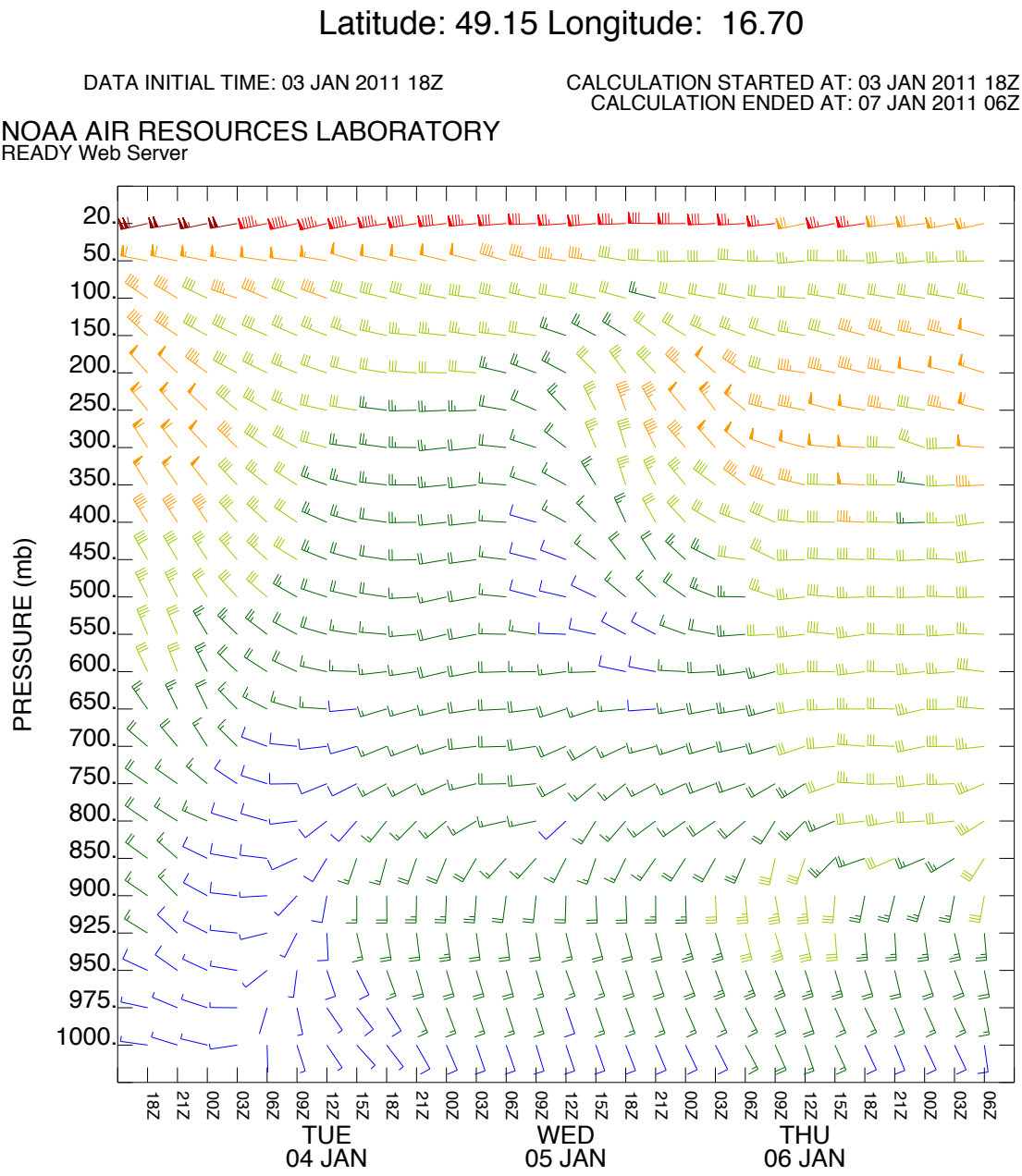
2.5 Vítr a termika

V této podkapitole se lehce odchýlíme od popisu balónového létání jako takového a podíváme se na roli proudění vzduchu v balónovém létání. Nastíníme postup, jak zjistit sílu a směr větru při průletu vrstvou. Pojmem vítr se nejčastěji označuje horizontální proudění vzduchu vznikající vyrovnáváním tlaku v atmosféře. Jak již bylo uvedeno výše, vítr může měnit v daném čase svoji rychlost a směr v závislosti na nadmořské výšce. Tohoto poznatku se při balónovém létání hojně využívá, problémem však zůstává jak "větrnou mapu"získat. Existují předpovědi povětrnostních podmínek, jsou ale poměrně nepřesné. NOAA - ARL (National Oceanic and Atmospheric Administration - Air Resource Laboratory) takovou předpověď poskytuje, viz obrázek 2.2. Různé větrné anomálie vznikají v blízkosti terénních nerovností, jako kopce a údolí, což asi není potřeba dále rozebírat.

Horkovzdušný balón v má horizontální rovině vzhledem ke svému tvaru poměrně veliký odpor prostředí. Není tedy úplně snadné jednoduchou úvahou stanovit sílu větru v jednotlivých vrstvách v závislosti na pohybu balónu. Minimálně pro primární účely otestování aplikace bude odpor prostředí položen maximu a vektor rychlosti balónu tak bude rovný vektoru pohybu okolního větru. Toto bude ve skutečnosti platit v případě, kdy balón aspoň nějakou dobu setrvá v dané letové hladině.

Otázkou zůstává, jak podrobně zvolit intervaly letových hladin, aby mělo měření dostatečnou přesnost a výpočet měl smysl. Pokud bude výška hladiny příliš malá nebo hladinou balón prostoupá příliš rychle, povede to k nepřesnostem ve výpočtu.

Podstatnou roli v balónovém létání hraje termika. Tento fyzikální jev se projevuje stoupavými proudy vzduchu a nastává při zahřátí zemského povrchu slunečním svitem. Různé typy povrchu absorbují sluneční záření rozdílně, čímž vznikají nesourodé proudy vzduchu. Nečekané proudy vzduchu mohou způsobovat při balónovém létání problémy v podobě náhlého stoupání či klesání. S ohledem na termiku startují balón buď brzy ráno, kdy krajina není ještě dostatečně prohřátá, aby byla termika aktivní, nebo pozdě večer, kdy už krajina chladne a termické proudy ztrácejí na své síle.



Obrázek 2.2: Větrná předpověď pro konkrétní místo a čas. Na svislé ose jsou vyznačeny letové hladiny označené atmosférickým tlakem. Vodorovná osa je časová. Standardní značky určují sílu (typ značky) a směr větru (natočení značky). Zdroj [16]

Kapitola 3

iPhone a iOS

V této kapitole bude přiblíženo, jakým způsobem se pro operační systém iOS použitý na platformě iPhone, vyvíjí aplikace. Vyjmenovány budou důležité aspekty poslední verze systému a poslední verze zařízení pro tuto práci. Uvedeny budou veškeré nutné (programové) vybavení pro vývoj, nasazení a testování. Dále metody a architektury programovaní pro iOS. Na závěr kapitoly prozkoumáme možné metody nasazení aplikace a testování.

3.1 Základní informace

Mobilní zařízení iPhone společnosti Apple Inc. spadá do kategorie smartphone. První verze byla uvedena na trh v červenci roku 2007, poslední verze, v pořadí čtvrtá, byla představena 24. července 2010. Nad rámec akcelerometru a dalších senzorů obsažených v původním modelu přidává poslední verze, označovaná jako *iPhone 4*, také A-GPS, digitální kompas a gyroskop. Zejména A-GPS patří mezi senzory, které jsou pro účel této práce nezbytnou záležitostí. Kompas, akcelerometr, gyroskop nebo 5 MPix fotoaparát by mohly přinášet také zajímavé informace pro výslednou aplikaci, jejich využití však není bezpodmíněčně nutné. Informace jsou zobrazovány prostřednictvím kapacitního dotykového multi-touch displeje s úhlopříčkou 9 cm a rozlišením 640x960 bodů. Interakce s uživatelem probíhá pomocí dotyků a dotykových gest. Přístroj je schopen zaznamenat až 5 samostatných dotyků současně, poklepání, tažení, cvrnknutí (švihnutí) a svírání nebo rozevírání prstů na displeji, terminologie převzata z oficiální uživatelské příručky, viz [14]. Kromě dotykového displaye se iPhone ovládá pomocí několika klasických tlačítek ukázaných na obrázku 3.1. Rozhraní přístroje je tvořeno WiFi, bluetooth a 30pinovým konektorem. Tyto parametry dávají společně široké možnosti a prostor při návrhu aplikace. Ostatní parametry *iPhone 4*, popř. předchozí verze 3GS je možné získat na stránkách výrobce, viz [7].

Operační systém *iOS* (dříve známý pod jménem *iPhone OS*) vyvinutý společností Apple Inc. na míru pro iPhone a další mobilní zařízení, jako iPod Touch a iPad. Je derivátem *Mac OS X* používaného pro stolní a přenosné počítače Macintosh. Jedná se tím pádem o unixový systém, jejichž společným základem je operační systém Darwin, který byl odvozen od systémů NeXTSTEP a BSD. Původní verze *iOS* byla předmětem kritiky pro nedostatečnou funkcionalitu, kterou konkurence již v té době standardně nabízela. Jako příklad uveďme záznam vidosekvence, *cut & paste*, multitasking atd. S každým novým modelem zařízení vychází pravidelně také nová verze iOS. Současná verze *iOS 4.3* již většinu nedostatků pokryla a jako každá hlavní verze přidává také množství nové funkcionality.

Obrázek 3.1: V horní části jsou vyznačeny ovládací prvky iPhone 4. Ve spodních obrázcích pak rozhraní přístroje. Zdroj [7]

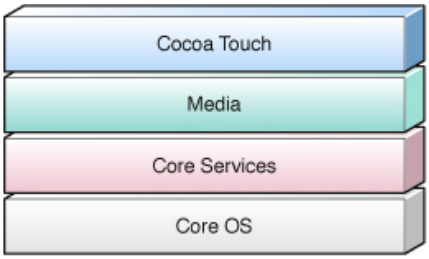
3.2 Programování pro iOS

Operační systém iOS se skládá ze čtyř vrstev zabezpečujících různou míru abstrakce a poskytujících služby. Pro iOS je možné vyvíjet dva typy aplikací – webové a nativní. Webové aplikace jako implementace HTML, CSS a JavaScriptu běží naserveru a uživatel s nimi interaguje pomocí předinstalovaného webového prohlížeče Safari. My se budeme zabývat vývojem nativních aplikacích, které jsou nainstalovány na zařízení a zobrazeny na základní obrazovce mezi systémovými aplikacemi jako Telefon, Obrázky, Kalendář atd. K tomu, abychom byli schopní vyvíjet vlastní aplikace, musíme pochopit jak základní architekturu systému tak i stavební struktuře jednotlivých aplikací. Podívejme se proto nejprve na uspořádání systému iOS.

3.2.1 Čtyři vrstvy

Jak již bylo zmíněno, systém se skládá ze čtyř základních vrstev. Na obrázku 3.2 je možné vidět jejich uspořádání.

Směrem nahoru po vrstvách budeme nacházet více a více sofistikovanější rozhraní implementované nejprve v jazyce *C*, později *Objective-C* umožňující používat systémové služby a prostředky. To nám dává na výběr jak implementaci provést. Obecně platí, že bychom měli používat vždy framework s nejvyšší možnou míru abstrakce, neboť vyšší vrstvy poskytují optimalizované objektově orientované abstrakce pro nízkoúrovňové konstrukce. Zjednodušují psaní kódu, redukují jeho množství a zapouzdřují komplexní problémy, jako práci s vlákny nebo sockety. Nicméně pokud má vývojář potřebu používat nízkoúrovňové techno-



Obrázek 3.2: Abstraktní vrstvy iOS. Nejvrchnější vrstva *Cocoa Touch* je nejpoužívanější a představuje rozhraní s nejvyšší mírou abstrakce. Na úplném dně leží základní vrstva *Core OS*, na které ostatní vrstvy stavějí. Zdroj [13]

logie nebo technologie, které nejsou dostupné z vyšších vrstev, nic mu nebrání použít framework s nižší úrovní abstrakce. Následuje přehled jednotlivých úrovní a služeb postupně dle míry abstrakce od nejvyšší po nejnižší.

Cocoa Touch je vrstva s nejvyšší abstrakcí a obsahuje klíčové frameworky pro stavbu každé aplikace. Definuje aplikační infrastrukturu a podporu pro klíčové technologie, jako multitasking, dotykové události, push notifikace, peer-to-peer služby a další. Touto vrstvou by se měl zabývat návrhář jako první. Přibližme nedávno představenou technologii iOS – Multitasking. Je dostupný od verze iOS 4.0 a umožňuje aplikacím pokračovat v běhu i v případě, kdy uživatel stiskne tlačítko pro návrat na hlavní obrazovku a aplikace je umístěna do pozadí. Běh aplikace je zpravidla přerušen, ale její kód je ponechán v paměti pro rychlé opětovné spuštění. Běh v pozadí je povolen pouze ve třech případech. Zaprvé když aplikace požádá o konečný procesorový čas na dokončení nějaké důležité úlohy, nebo pokud se aplikace deklaruje jako nezbytná pro provoz některé služby, která vyžaduje pravidelný běh na pozadí, anebo zatřetí chce-li aplikace používat lokální notifikace a upozornění, ať už aplikace běží nebo nikoliv. Hlouběji dokumentace nebude rozebírána. Více detailů o této a ostatních vrstvách je možné získat v oficiální dokumentaci na [13]. Za zmínku by stál framework *UIKit*, který je také součástí vrstvy *Cocoa Touch* a je společným základem pro všechny grafické aplikace, ovládané dotykovými událostmi.

Další v pořadí následuje vrstva *Media*. Obsahuje technologie zabezpečující grafiku, zvuk a video. Nejjednoduší a nejefektivnější způsob jak zabezpečit grafickou stránku aplikace, je použít standardní prvky GUI z vyšší vrstvy. Někdy to ale nemusí stačit a je potřeba použít některou technologii pro vykreslení grafického kontextu. Tyto metody jsou *Core Graphics* neboli *Quartz* pro zabezpečení 2D rendrování, *Core Animation* poskytující animaci prvků rozhraní, *OpenGL ES* k hardwarové akceleraci 2D a 3D scény a další. Audio technologie zabezpečují přehrávání a záznam zvuku a ovládání vibrací mobilního zařízení. Video technologie této vrstvy zabezpečují záznam a přehrávání videa ať už z lokálního souboru nebo streamování ze sítě. Audio i video technologie nabízí několik frameworků, které se liší mírou abstrakce a tedy i složitostí použití.

Třetí nejvyšší vrstva se nazývá *Core Services*. Osahuje základní systémové služby, které používají všechny aplikace. Mimo tyto systémové služby sem spadá podpora databáze *SQLite* bez nutnosti odděleného databázového serveru nebo knihovna pro parsování XML dokumentů.

Poslední, nejnižší vrstva *Core OS* poskytuje nízkoúrovňové služby, na kterých je většína ostatních technologií postavena. I pokud vývojář nepotřebuje přímo používat frameworky této vrstvy, stejně je zpravidla využívá nepřímo prostřednicvím vyšších vrstev. Základ tvoří

systémové knihovny zapouzdřující prostředí jádra, ovladače a zabezpečující UNIXové rozhraní operačního systému nízké úrovně. Rozhraní této vrstvy implementovaná v jazyce C poskytují programový přístup k vláknům, síťovým schránkám, souborovému systému, matematickým operacím, správě paměti atd. Kromě systémových knihoven do této vrstvy náleží např. také framework spravující komunikaci s externími zařízeními připojenými přes bluetooth nebo 30-pinový konektor.

3.3 Aplikační prostředí Cocoa Touch

Cocoa Touch (resp. jeho předchůdce Cocoa) jako aplikační prostředí iOS (resp. Mac OS X) je tvořeno množinou objektově orientovaných frameworků, runtime prostředím a integrovaným vývojovým prostředím. Jako každé aplikační prostředí má *Cocoa* runtime stranu, kde *Cocoa* prezentuje uživatelské rozhraní a úzce spolupracuje s viditelnými částmi systému nad rámec aplikace jako např. stavovým řádkem, a vývojovou stranu, která sestává ze souboru znovu použitelných a přizpůsobitelných tříd umožňujících velmi rychlý vývoj aplikací. Za zmínku stojí dva základní frameworky *Foundation* a *UIKit*, které jsou nezbytnou součástí každé aplikace.

3.3.1 Jazyk

Při vývoji prostřednictvím Cocoa je možné použít několik programovacích jazyků. Základním jazykem však vždy zůstává *Objective-C*, který je nadmnožinou *ANSI C*, oproti kterému je rozšířen o některé syntaktické a sémantické charakteristiky na podporu objektové orientace. Není tedy problém do kódu v *Objective-C* vkládat volně kusy kódu psaného v *ANSI C* nebo i *C++*. Popis syntaxe, sémantiky a dalších vlastností jazyka *Objective-C* jako správa paměti apod. není předmětem tohoto dokumentu, k programování pro platformu *iOS* jsou však tyto znalosti nezbytné. Podrobě se s nimi lze seznámit např. v oficiální příručce, viz [17].

3.3.2 Návrhové vzory

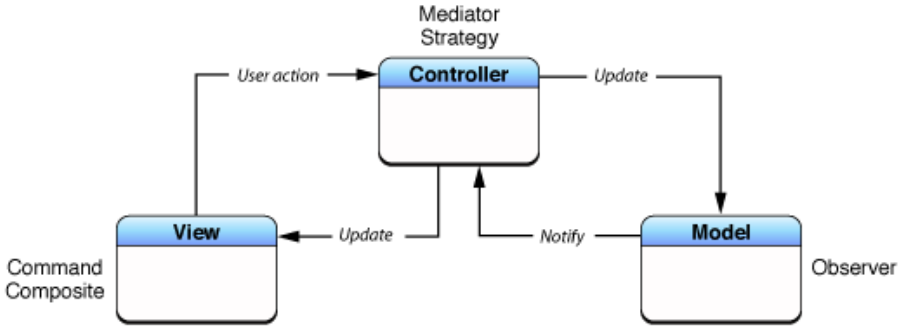
Mnoho konstrukcí v Cocoa využívá navrhových vzorů, které poskytují řešení opakujících se problémů v opakujících se situacích. Vzory jsou jakési šablony pro konkrétní návrhy nebo jinak, konkrétní návrhy jsou instancí návrhových vyzorů. Více o návrhových vzorech v [3].

Nejdůležitějším návrhovým vzorem k pochopení struktury aplikace je *MVC (Model-View-Controller)*. Tento vzor definuje, jak napovídá jeho název, tři typy objektů. Jeho hlavní účel je umožnit recyklaci objektů a zásah do kterékoliv z nich bez potřeby upravovat jiné. Podívejme se nyní blíže na jednotlivé role objektů. Komunikace mezi objekty MVC je vyobrazena na obrázku 3.3.

- *Model* zapouzdřuje data a definuje logiku pro jejich manipulace. Všechna persistentní data aplikace by měla být součastí pouze objektů tohoto typu. Reprezentují znalost domény a měly by být opětovně použitelné. Objekty modelu by neměly být jakkoliv propojeny s uživatelským rozhraním a neměly by rozhodovat o tom, kde, kdy a jakým způsobem budou obsahující informace vykreslovat. Výjimkou budiž objekty reprezentující grafický obsah, které by měly umět samy sebe vykreslovat, nicméně by neměly určovat, kde se tomu stane.

- Typ *view* reprezentuje objekty uživatelkého rozhraní, které zobrazují informace o doméně. Teoreticky by objekty typu view neměly ukládat data, která zobrazují. Tyto objekty nabývají v rámci Cocoa různých podob kupř. *UIButton, UILabel, UITextField* nebo *UIView* plnící funkci kontejneru pro jiné objekty tohoto typu. Data mohou čerpat z jednoho nebo více objektů modelu zároveň. Rozhraní je typicky uloženo v souboru typu *NIB*, což je v podstatě archiv objektů uživatelského rozhraní.

- *Controller* trojici doplňuje v roli prostředníka mezi modelem a view. Controller je zodpovědný za přístup objektů typu view k informacím modelu, které mají zobrazovat, a ujišťují se, že reagují na změny v modelu. Controller zaopatřuje interakci s uživatelem. Např. vyhodnocuje změnu hodnoty nebo volbu skrze view a komunikuje tuto informaci modelu, popř. upravuje samotné view. Controller, jelikož jej vytváří, je zpravidla vlastníkem svého odpovídajícího view definovaného souborem *NIB* popř. *XIB* s uživatelským rozraním .



Obrázek 3.3: Schéma komunikace v iOS dle vzoru Model-View-Controller. Objekty spadající do skupiny *View* nevstupují do interakce s modelem aplikace. Zdroj [13]

Komunikace mezi objekty je zabezpečena pomocí vzájemných referencí. Základním typem reference je vývod (*Outlet*), který je definovaný pomocí nástroje Interface Builder pro tvorbu uživatelského rozhraní. Obvykle bývá vytvořeno spojení mezi prvky uživatelského rozhraní a odpovídajícím controllerem. Ještě před propojením outletů v Interface Builderu je potřeba v hlavičkovém souboru controlleru tyto outlety deklarovat. Reference jsou ustanoveny v momentě, kdy je zpracován *NIB* soubor. Další obvyklá metoda komunikace probíhá pomocí objektů typu *Delegate* a jim podobných *Data Source*, podrobněji viz [13].

3.4 Nástroje

Aplikace pro iOS je možné oficiálně vyvíjet pouze na počítačích Macintosh s procesory Intel. Z programového vybavení potřebujeme integrované vývojové prostředí *Xcode* doplněné nástroji *Interface Builder* a *Instruments*. Poslední dva uvedené nejsou pro vývojáře nezbytně nutné, zpravidla jsou ale jeho velmi oblíbenými pomůckami.

Xcode umožňuje mj. správu projektu, psaní kódu, kompilaci a konstrukci spustitelných souborů, debugging a správu repozitáře. Obsahuje několik použitelných šablon aplikací, které jsou bez jakéhokoliv zásahu spustitelné a funkční, což je jistě dobrý odrazový můstek pro začínajícího vývojáře. *Xcode* přehledně organizuje zdrojové soubory a veškerá nastavení týkající kompilace. Mimo jiné si lze zvolit, za se má spustitelný kód sestavit pro simulátor,

který je součástí IDE, nebo zařízení, které je možné připojit k počítači a ladit na něm aplikaci za asistence debuggeru. Simulátor má ale pouze omezené schopnosti. Nepodporuje některé hardwarové prvky jako akcelerometr, A-GPS, telefonii apod. Ve verzi Mac OS 10.6 Snow Leopard však iPhone Simulator disponuje jakousi omezenou podporou GPS. Existuje nástroj třetí strany nazvaný *iSimulate* viz [19], který má dvě části. Jednak programovou knihovnu, která se připojuje k vybranému projektu, a jednak aplikaci pro zařízení iPhone. Zkompilujeme-li projekt s touto knihovnou a spustíme-li jej v simulátoru, můžeme se k běžící aplikací připojit z aplikace na mobilním zařízení. Počítač, na kterém je spuštěn simulátor, a iPhone musejí být spolu v síti. Ze zařízení jsou pak přenášena data ze senzorů jako akcelerometr, A-GPS a dotekové události na displeji přímo do aplikace v simulátoru. Tento nástroj je užitečný, potřebujeme-li testovat aplikaci závislou na některém z uvedených senzorů a nechceme-li ztrácet čas opakovanou instalací na zařízení.

Interface Builder je nástroj umožňující vizuálně sestavovat uživatelské rozhraní metodou drag & drop, konfiguraci použitých komponent a uložení výsledku do *NIB* souboru. Tento archiv obsahuje všechny nutné informace, které *UIKit* potřebuje pro rekonstruci uživatelského rozhraní za běhu aplikace. Spravuje také propojení s existujícími objekty aplikace, jak bylo uvedeno výše.

Aplikace *Instruments* pomáhají analyzovat vytvořenou aplikaci z výkonostního hlediska ať ji testujeme v simulátoru nebo na zařízení. Získáváme informace o využití paměti, aktivitě pevného disku, síťové aktivitě a grafickém výkonu. Data přehledně prezentovaná grafy je možné ukládat a později porovnávat během optimalizace aplikace.

Nutnou součástí pro vývoj pro danou platformu je SDK (*Software Development Kit*), které obsahuje dokumentaci, zdrojové kódy, ukázkové aplikace, nástroje, knihovny a další součásti použité při vývoje aplikace. Současnou verzí je iOS SDK 4.2, vždy stejná jako verze iOS.

3.5 Nasazení

Oficiální metoda instalace výsledné aplikace na zařízení sestává z několika kroků. Nejprve se vývojář zaregistruje jako *Apple Developer*, na [13] a dostane přístup k potřebným vývojovým nástrojům, viz podkapitola 3.4. V této fázi je možné vytvořené aplikace testovat simulátoru. Testování aplikací mimo simulátor a přístup k oficiální vývojářské komunitě je podmíněno účastí na některém z vývojářských programů označovaných *iOS Developer*. Dobrý startovací bod představuje např. kniha [4].

- *iOS Developer Program* pro fyzické osoby a společnosti umožňuje nad rámec společných vlastností placených programů distibuci jednak *ad-hoc* a jednak prostřednictvím *App Store*, viz níže na straně 15. Cena tohoto programu je 99$ za rok.
- *iOS Developer Enterprise Program* za cenu 299 $ za rok určený pro podniky sice neumožňuje šíření prostřednictvím *App Store*, ale dovoluje vnitropodnikovou distribuci.
- *iOS Developer University Program* je zdarma a je určen pro vzdělávací instituce. Dovoluje tvořit vývojové týmy až o 200 studentech a využívat všechny možnosti testování aplikací na zařízení. Program bohužel nepodporuje distribuci aplikací.

App Store – on-line obchod s aplikacemi pro iOS – společnosti Apple Inc. byl uveden do provozu 10. července 2008. Tento revoluční nástroj slouží ke snadné distribuci aplikací nejen spol. Apple Inc., ale i třetích stran koncovému uživateli. V současnosti *App Store*

obsahuje přes 300 000 aplikací. Po zprovoznení *App Store* přišla většina významných konkurenční platforem také s vlastní verzí on-line obchodu s aplikacemi. Své vlastní aplikace může vývojář zdarma poskytovat nebo prodávat v *App Store*, 30% výdělku vývojáře ale náleží spol. Apple Inc.

Existuje i neoficiální, pololegální metoda distibuce, která vyžaduje úpravu iOS, která povoluje přístup ke všem prostředkům systému, tzv. *Jailbreak*. Legálnost této úpravy je předmětem diskuzí, nicméně zařízení podle výrobce pozbývá záruku. Kompilace aplikace s profilem *Device* vyžaduje podepsání kódu certifikátem, který vývojář obdrží společně s registrací v jednom z placených programů. Podepisování je však možné obejít a instalovat aplikace na odemčené zařízení pomocí vzdáleného repozitáře nebo přímo kopií přes *ssh* spojení. Jelikož je tento způsob instalace právně pochybný, nebudou zde uvedeny žádné konrétní návody, zdroje ani postupy.

Kapitola 4

Návrh aplikace

Tato kapitola je souborem konkrétních informací potřebných k implementaci aplikace. Popisuje proces získávání a zpracování některých z nich. Nejprve jsou rozebrány funkční požadavky vyplývající ze zadání a z konzultace s balónovými piloty. Grafické uživatelské rozhraní je v tomto případě velmi důležitým prvkem řešení, a proto na něj byl kladen mimořádný důraz podpořený dotazníkem pro balónové piloty, aby bylo dosaženo výsledků vyhovujících provozním podmínkám. Dotazník byl z části směřován i na funkčnost aplikace, ale jeho závěry mají dopad spíše na uživatelské rozhraní aplikace. Z nasbíraných informací je vytvořen model logika přechodů obrazovek a konečná struktura aplikace, což je popsáno v poslední podkapitole.

4.1 Požadavky na funkčnost

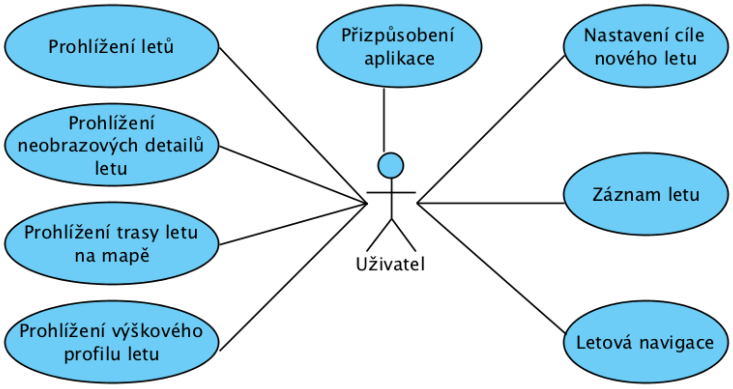
Požadavky na funkčnost aplikace vycházejí jednak ze zadání práce a jednak z praxe neboli z požadavků balónových pilotů na takovou aplikaci. První úlohou je vedení pilotního deníku tzv. *logbook*, což spočívá v zaznamenávání a prezentaci informací o provedených letech. Sběr těchto informací probíhá pomocí vestavěných senzorů mobilního zařízení (*A-GPS*) a zadáváním textu na virtuální klávesnici přístroje přímo od pilota. Konkrétně jsou to následující informace. U každé z nich je uvedeno, jestli informaci zadává uživatel nebo je zajištěna programově. Výčet vychází také z dotazníku rozebraném v podkapitole 4.3.

- Celkový čas letu (iOS),
- celková vzdálenost (iOS),
- maximální rychlost (iOS),
- průměrná rychlost (iOS),
- maximální vertikální rychlost (iOS),
- průměrná vertikální rychlost (iOS),
- maximální a minimální nadmořská výška (iOS),
- trasa letu (iOS),
- směr a síla větru v závislosti na nadmořské výšce a času (iOS),

- vertikální průběh letu (iOS),
- rozmezí směru větru (iOS),
- posádka letu (uživatel),
- spotřeba paliva (uživatel),
- podmínky letu a další poznámky (uživatel).

Z uvedených požadavků nebudou implementovány všechny, ale pouze ty, které byly shledány jako důležité, viz 4.3. Druhý úkol aplikace je navigační povahy. Je jím pomoc pilotovi vést balón směrem k určenému cíli v reálném čase. Tuto stranu aplikace je možné shrnout do následujících dílčích bodů, které jsou zabezpečeny výhradně systémovými prostředky a uživatel pouze přijímá vyhodnocené a zpracované informace.

- Aktuální pozice vyobrazená na mapě,
- směr pohybu,
- rychlost pohybu balónu,
- aktuální směr a síla větru v závislosti na nadmořské výšce,
- aktuální nadmořská výška,
- směr k zadanému cíli.



Obrázek 4.1: Diagram případů užití jediného uživatele v systému. Případy, které spolu souvisejí, jsou vždy uvedené ve sloupci pod sebou.

Vzhledem k rozsahu této práce a k potenciálnímu objemu výsledného produktu nebylo možné implementovat vše a bylo nutné vybrat funkčnost, která je podstatná pro fungování aplikace a je zajímavá z pohledu řešení a využitelnosti. Aplikací typu deník, které pouze zaznamenávají a prezentují data, jako místo vzletu a přistání, délka letu, posádka, spotřebované palivo atd., existuje několik a v podstatě se mnoho neliší od obyčejného papírového deníku. Proto oblasti funkčnosti, na které je tento projekt zaměřen, vyžadují netriviální sběr, výpočet a nebo zobrazení dat. Jedná se např. o zobrazení trasy na mapě, výškový

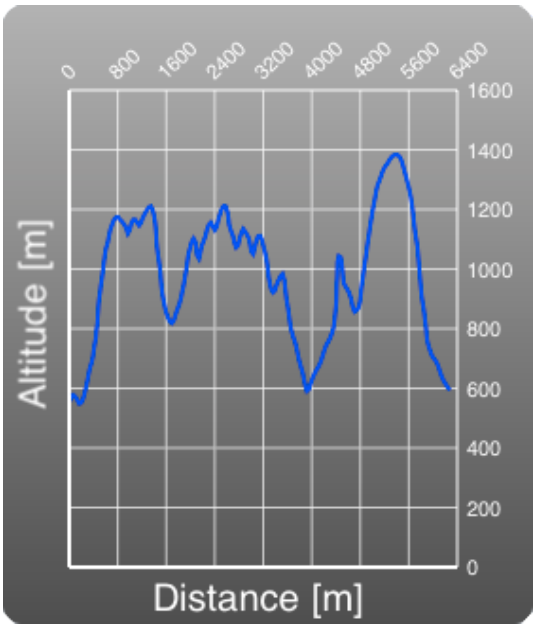
profil letu a zejména o navigaci stavějící na získání informací o síle a směru větru v závislosti na nadmořské výšce a jejich přehledném a přizpůsobivém zobrazení. Souhrn funkčních požadavků na aplikaci je zobrazen pomocí diagramu případů užití na obrázku 4.1. Vzhledem k charakteru systému je prozatím uvažována pouze jedna jediná role uživatele.

4.2 Prvky grafického uživatelského rozhraní

Následující podkapitola popisuje dva typy prvků uživatelského rozhraní. První jsou standardní prvky *iOS* a druhé jsou prvky navržené na míru potřebám aplikace tak, aby vyhovovaly jejím požadavkům. Většinou bylo u každého prvku navrženo více variant a ta, která byla zvolena piloty prostřednictvím dotazníku jako nejlepší, byla implementována. Jako barvy grafického uživatelského rozhraní byly zvoleny bílá, modrá a šedá v různých odstínech, jelikož jsou to barvy, do kterých lze přizpůsobit většinu standardních prvků, jako lišty, tlačítka a tabulky.

4.2.1 Standardní prvky

Ve všech případech, kdy to bylo možné, jsou využity standardní prvky systému *iOS*, na které je uživatel mobilního zařízení *iPhone* zvyklý a ví, jak s nimi pracovat. Každý je laděn do výše zmíněných barev, aby prostředí aplikace působilo jako důvěrně známé, čisté a jasné. Konkrétně se jedná o tabulku nebo chcete-li seznam provedených letů, textové informace o detailu letu, tlačítka napříč celou aplikací, segmentovaná tlačítka pro přizpůsobení aplikace a mapy. Standardní prvky samostatně nejsou příliš poutavou kapitolou, obsáhlejší bude jejich složení a propojení ve výslednou struktu aplikace.

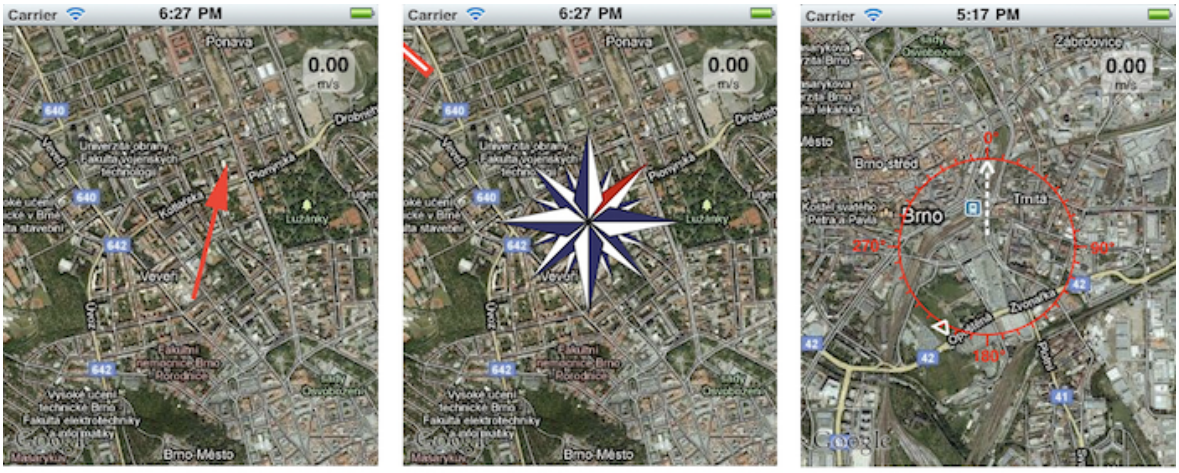


Obrázek 4.2: Výškový profil letu. Rostoucí vzdálenost je vyznačena na vodorovné ose. Závislá veličina — nadmořská výška — je na svislé ose.

4.2.2 Prvky rozhraní na míru

V této podkapitole budou přiblížena místa grafického uživatelského rozhraní, kde by nebylo možné vystačit s běžnými články systému. Jsou to prvky, které úzce souvisejí s doménou aplikace. Abychom následovali strukturu tak jako vždy, podívejme se nejprve na část pilotního deníku. Pro účely logování lze vystačit se standardními prvky až na zobrazení výškového profilu dráhy letu. Nativní prostředky pro kreslení grafů vývojové prostředí neobsahuje. Na obrázku 4.2 je znázorněn navržený jednoduchý graf závislosti nadmořské výšky na uražené vzdálenosti, který koresponduje se zvoleným barevným schématem.

Celá navigace by v ideálním případě měla probíhat se zobrazenou mapou aktuálního okolí, se kterou musí být veškeré prvky úzce spjaty. Nesmějí zabírat velikou plochu na displeji, aby zbytečně nepřekrývaly prostor mapy. Navigační část zahrnuje dvě položky, které bylo nutné přizpůsobit. Jednak informaci o stavu balónu v daný okamžik obsahující současnou polohu, směr a rychlost pohybu a směr k cíli, pokud byl zadán. Jednak směr a sílu větru pro jednotlivé letové hladiny.



Obrázek 4.3: Zvažované možnosti zobrazení současného stavu balónu. Vlevo jednoduchá varianta s šipkou, uprostřed navigační růžice a vpravo kružnicí s šipkou. Zobrazení GPS souřadnic textově by bylo prostorově úsporné, ale text neposkytuje dostatečnou názornost.

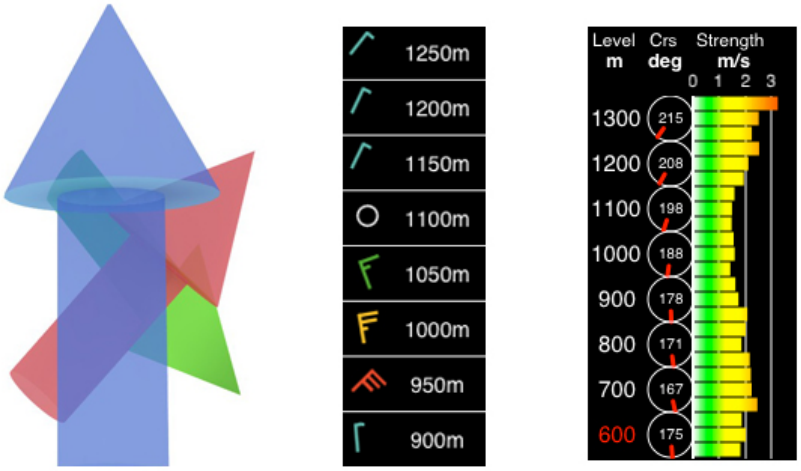
- *Šipka* — pro ilustraci na obrázku 4.3 vlevo — představuje jednoduchou možnost, jak označit současnou polohu počátkem šipky, směr a rychlost pohybu natočením a délkou šipky. Mezi nevýhody tohoto řešení patří nepřesnost určení rychlosti pohybu a také nemusí být uživateli zcela zřejmé označení aktuální polohy. V případě, že uživatel zadá cíl letu, bylo by potřeba přidat ještě jednu šipku označující, kde se vzhledem k pozici balónu nachází.

- Další zvažovanou reprezentací je *směrová růžice* — na obrázku 4.3 uprostřed — používaná v kompasech, která by poskytla údaj o poloze a barevným odlišením jednoho z paprsků i směr letu. Informace o rychlosti by musela být doplněna textově a kurs na cíl znázorněn např. pomocí dalšího prvku na okraji displeje. Nevýhodou je fakt, že grafika zabírá relativně hodně prostoru a překrývá oblast mapy, ve které se pilot právě nachází.

- *Kružnice* — na obrázku 4.3 vpravo, někdy také označována jako růžice — oproti růžici

v předcházejícím bodu dokáže pojmout i údaj o směru na cíl, ale také musí být doprovázena textovým údajem o rychlosti. Rychlost by se dala stejně jako u první varianty vyjádřit délkou šipky, bylo by to však velmi nepřesné. Tato grafická reprezentace překrývá pouze malé množství podkladu a poskytuje přesnější směrové informace než předchozí možnost díky vyznačení drobných intervalů po obvodu. V tomto případě se vhodně jevil interval 10 stupňů.

Všechny výše zmíněné varianty byly v rámci dotazníku na uživatelské rozhraní nabídnuty balónovým pilotům, kteří zvolili subjektivně nejpraktičtější možnost. S přihlédnutím k výsledkům, které jsou rozvedeny v podkapitole 4.3 věnující se dotazníku, byla jako nejvhodnější vybrána poslední varianta s kružnicí. Jelikož je textová informace přesná a prostorově nenáročná, bude uvedena společně s obrazovou.



Obrázek 4.4: Grafické reprezentace směru a síly větru. Vlevo varianta s prostorovými šipkami, uprostřed tabulka s letovými hladinami opatřenými standardními značkami a vpravo opět tabulka s letovými hladinami doplněná směrovými kružnicemi a diagramem znázorňujícím sílu větru.

Na obrázku 4.4 jsou tři zvažované grafické reprezentace směru a síly větru v závislosti na nadmořské výšce. Vítězná varianta by měla splňovat stejné náležitosti jako v předchozím případě, tj. prostorovou skromnost, názornost a v této situaci i nově — jak později zjistíme — schopnost interakce s uživatelem. Tento prvek tvoří rozhraní těžiště celé aplikace, bylo mu proto věnováno mnoho pozornosti a byl také hlavním důvodem sběru informací z praxe od balónových pilotů v podobě dotazníku.

- Nejvíce vlevo je nejodvážnější alternativa s *prostorovými šipkami* umístěnými nad sebou. Na mapovém podkladu by každá z nich byla pomyslně umístěna v odpovídající letové hladině. Natočení šipky udává směr větru, barva pak jeho směr. Šipky zabírají docela velikou část mapy, ale jejich průhlednost odkrývá podklad téměř v plném rozsahu. Za nevýhodu lze považovat nepřesnost prezentovaných informací. Výhodou této varianty je grafická atraktivita.

- Uprostřed na obrázku 4.4 je řešení pomocí standardních značek zavedených v tzv. *staničním modelu*, který se používá pro popis meteorologické situace v místě meteorologické stanice vydaný Světovou meteorologickou organizací [21] (*WMO*). Každé letové

hladině odpovídá jedna značka a každá značka označuje rychlost větru v určitém intervalu. Více o staničním modelu se lze dozvědět např. v pubikaci *Meteorology Today* [1] v příloze B. Natočení značky určuje směr větru a u každé značky je údaj o nadmořské výšce určující letovou hladinu. Tento model poskytuje poměrně přesné údaje, nezabírá příliš velikou plochu displeje a je možné jej relativně rychle číst. Jeho velikost navíc dovedeme redukovat přidáním posuvníku, čímž snížíme počet zobrazených řádků tabulky a tím i celkovou přehlednost situace. Aktuální nadmořská výška může být v tabulce vyznačena např. odlišnou barvou řádku.

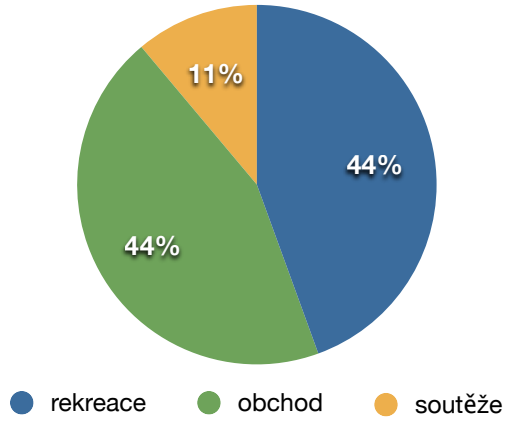
- *Graf* na obrázku vpravo je poslední navrženou alternativou. Skládá se ze tří sloupců. První sloupec určuje letovou hladinu. Druhý obsahuje informaci o směru větru zakreslenou kružnicí s ručičkou vhodnou pro rychlé rozhodování a textový údaj uprostřed s přesnou hodnotou ve stupních. Poslední sloupec tvoří graf závislosti rychlosti větru na letové hladině. Každou letovou hladinu pokrývá trojice údajů o rychlosti s vyšším výškovým rozlišením. Hodnoty v grafu jsou opět pro rychlé rozhodování podbarveny zeleno-žluto-červeným lineárním gradientem. Odečítání konkrétních rychlostí upřesňují vyznačené významné hodnoty na vodorovné ose. Graf by zabral ze všech zmíněných variant pravděpodobně nejvíce pixelů dispeje, ale stejně jako v předchozím případě řešení do jisté míry představuje použití posuvníku.

Stejně jako v případu s vizualizací aktuální polohy i tentokrát bylo přihlédnuto k závěrům dotazníku. Implementována byla poslední uvedená varianta. Diskuze nad touto volbou bude součástí následující kapitoly. Tímto jsou pokryta všechna sporná místa uživatelského rozhraní.

4.3 Dotazník pro balónové piloty

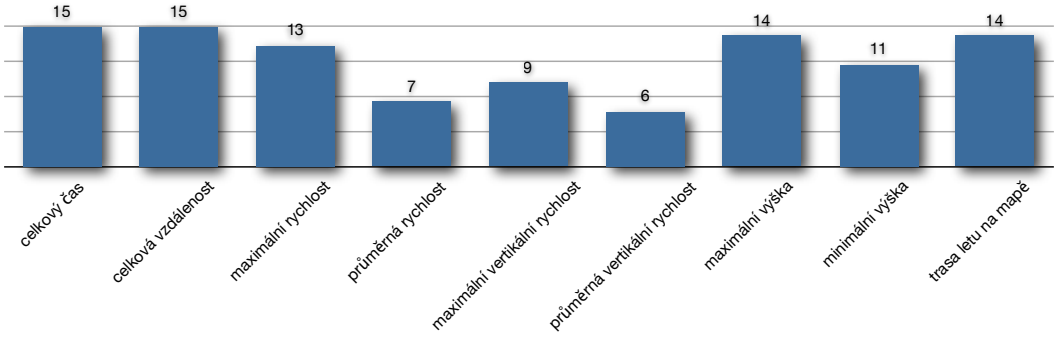
Následující kapitola se věnuje prostředku, který byl použit pro získání konkrétních informací nutných pro využitelnost aplikace v praxi. Dalším užitečným aspektem bylo navázání kontaktů důležitých pro hodnocení a testování aplikace. Jelikož balónoví piloti jsou balónovými piloty převážně ze záliby a nadšení, seznámení s touto prací bylo pro některé velmi atraktivní. Přislib další spolupráce napovídá zkvalitnění aplikace a případnému pokračování vývoje za rozsah této práce. Dotazník obsahuje čtyři části. První dobrovolná část zjišťovala výše zmíněný zájem pilotů a jejich vybavenost mobilní platformou *iPhone* pro potenciální testování aplikace v ostrém provozu. Druhá část zjišťovala profesní náležitosti a zkušenosti respondentů, které by bylo možné promítnout na váhu ostatních odpovědí. Názor zkušenějšího pilota by byl hodnotnější než názor nováčka. Třetí část zjišťovala vesměs nezbytný výčet informací obsažený v pilotním deníku. Konečně čtvrtá část zjišťovala požadavky na grafické rozhraní aplikace a na velikost intervalu letových hladin. Poslední skupina otázek byla vzhledem k jádru práce nejdůležitější a měla na výsledek největší vliv. Níže nejsou vyhodnoceny všechny otázky, ale jen ty směrodatné. Celý dotazník v plném znění tak, jak byl předložen pilotům, je přiložen jako příloha A. Vyplnilo jej 15 balónových pilotů.

Z prvních dvou částí stojí za zmínku rozložení příčin pilotování respondentů znázorněné v grafu na obrázku 4.5. Průměrná délka zkušeností s pilotováním horkovzdušných balónů se u tázané skupiny pohybovala lehce nad hodnotou 7 let. Výstupem třetí sekce je soubor informací zaznamenávaných během každého letu, který je zohledněn v podkapitole 4.1 zabývající se požadovanou funkčností. Počet hlasů jednotlivých variant znázorňuje



Obrázek 4.5: Převažující důvody pilotování všech respondentů. Většina z nich létá s balóny z obchodních a rekreačních důvodů.

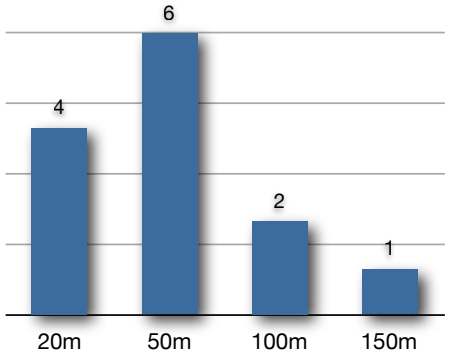
graf na obrázku 4.6. Nejdůležitější informace jsou podle předpokladů: celkový čas, celková vzdálenost, maximální výška, trasa letu na mapě a maximální horizontální rychlost.



Obrázek 4.6: Požadavky pilotů na soubor informací, které mají být zaznamenávány do pilotního deníku.

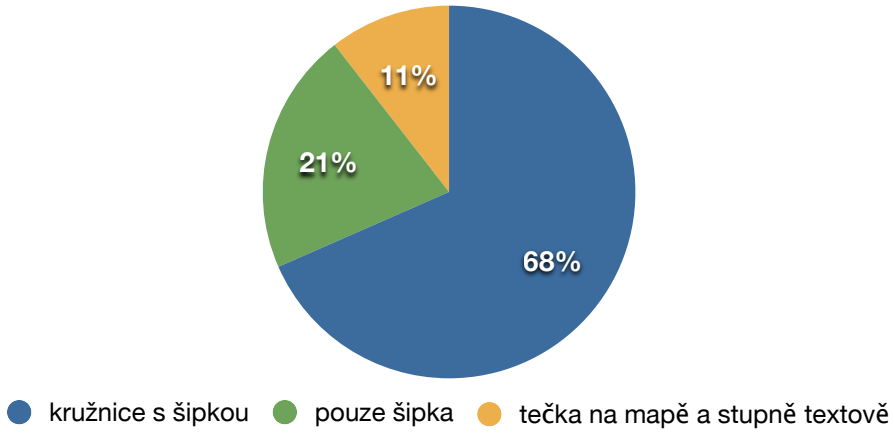
Poslední část poskytla nejužitečnější výstup a nejvíce ovlivnila celou aplikaci. Konkrétně většina pilotů uvedla, že raději zadávají cíl letu pomocí GPS souřadnic než na mapě. Před soutěžním letem totiž obdrží zadání letu včetně přesných GPS souřadnic cíle, kdežto cíl rekreačního letu není podstatný. Zadávání cíle na mapě takřka nemá využití. Dalším podstatným aspektem bylo zvolení vhodného intervalu letových hladin, proto byla tato otázka také do dotazníku zařazena. V grafu na obrázku 4.7 převažuje interval o velikosti $50\ m$, pro který se rozhodla zhruba třetina dotázaných. Ostatní odpovědi směřovaly na jiné intervaly a dva piloti explicitně uvedli, že by preferovali proměnnou velikost intervalu. Z toho důvodu byla přidána možnost volby intervalů dynamicky během letu. Tento fakt musejí respektovat prvky uživatelského rozhraní uvedené v podkapitole 4.2. Způsob volby intervalu bude uveden níže v kapitole 5 o implementaci.

Poslední dvě otázky zjišťovaly názory pilotů na prvky uživatelského rozhraní a pomohly z navrhnutých variant vybrat ty nejvhodnější. Jednotlivé návrhy jsou uvedeny v předchozí podkapitole. Piloti měli vždy vybrat jednu nebo dvě z nabízených alternativ. Rozložení preferencí první otázky (v dotazníku pod číslem 13) týkající se zobrazení aktuální polohy a směru letu ukazuje graf na obrázku 4.8. Jednoznačně zvítězila kružnice s šipkou. Druhá



Obrázek 4.7: Rozložení odpovědí pilotů na otázku, která zjišťovala ideální interval jednotlivých letových hladin. Větší část z nich se přikláněla k drobnějšímu dělení.

otázka v dotazníku s číslem 15 zjišťovala, jakým způsobem zobrazovat sílu a směr větru v letových hladinách. Možnost s podrobným diagramem pro rychlost získala většinu hlasů, viz graf na obrázku 4.9. Vizualizaci otázek lze dohledat v předchozí kapitole nebo v plném znění pod výše uvedenými čísly v příloze A.

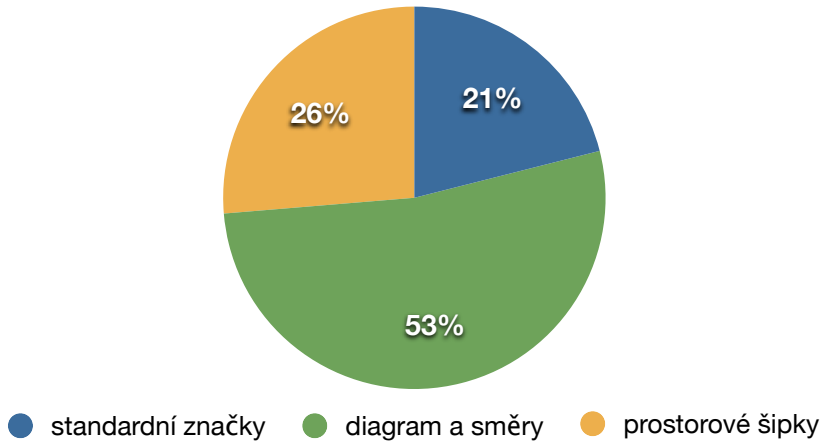


Obrázek 4.8: Přání pilotů na zobrazení aktuálních GPS informací z nabídnutých variant.

Shrneme-li nejdůležitější informace získané pomocí dotazníku, budou to tyto: seznam nutných údajů pro pilotní deník, potřeba variability intervalu letových hladin a výběr konkrétních prvků uživatelského rozhraní vyhovujících koncovým uživatelům.

4.4 Výsledná struktura

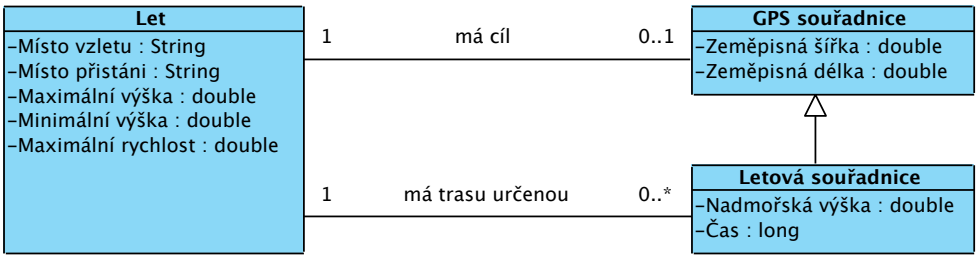
Následující řádky popisují výsledný návrh aplikace vycházející z informací a požadavků shromážděných v předchozích odstavcích této kapitoly. Uvedeme reprezentaci domény aplikace, popis jednotlivých obrazovek aplikace, logiku jejich přechodů a také diagram sekvence záznamu letu. Jelikož se struktury reprezentují lépe pomocí diagramů, bude v této kapitole převažovat obrazová informace nad textovou. Nejprve bude návrh popsán z obecnějšího pohledu, dále budou přiblíženy konkrétní použité prvky při implementaci.



Obrázek 4.9: Poměr odpovědí říkající preference zobrazení síly a směru větru.

4.4.1 Třídy

Model aplikace je vzhledem k jejím požadavkům poměrně prostý. Na obrázku 4.10 je znázorněn pomocí diagramu tříd. Obsahuje dvě třídy, kterými jsou let a souřadnice. Souřadnice obsahuje atributy se zeměpisnou šířkou a délkou, nadmořskou výšku a časové razítko. Let, který je ve vztahu *0* až *n* s třídou Letová souřadnice, obsahuje informace o místu vzletu a přistání, cíl letu, celkový čas, maximální a minimální výšce a maximání rychlosti.



Obrázek 4.10: Diagam tříd modelu. Třída Let obsahuje kolekci typu GPS souřadnice.

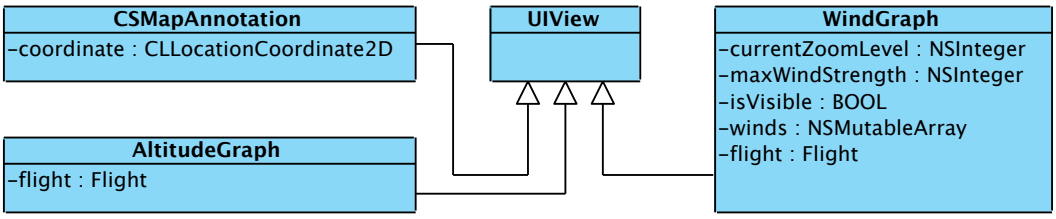
Třída Let by mohla obsahovat přímo atributy se zeměpisnou délkou a zeměpisnou šířkou cíle, jelikož je s třídou GPS souřadnice ve vztahu *1 – 0..1*, ale kvůli generalizačnímu vztahu letové a GPS souřadnice je toto vyřešeno samostantou třídou. Atributy třídy Let obsahující extrémy se mohou zdát redundantní, jelikož jsou algoritmicky vypočitatelné z letových souřadnic. Bylo by však nevýhodné je pokaždé, když jsou potřeba, počítat, protože výkon je na mobilním zařízení vždy drahý a jejich uložení spotřebuje minimum prostoru v paměti. Atributy s místy přistání jsou typu *string*, nikoliv *double*, protože obsahují adresu, jméno města nebo okresu. Souřadnice těchto míst lze bez výpočtu získat z první a poslední souřadnice letu.

Skupinu označenou *View* tvoří převážně standarní *API* třídy. Pro některé grafické prvky popsané v podkapitole 4.2 bylo potřeba navrhnout vlastní komponenty, které své vlastnosti dědí od třídy *UIView*. Konkrétně jsou to třídy obsažené v diagramu tříd na obrázku. Třída *CSMapAnnotation* vykresluje směrovou kružnici (růžici) přes mapu během navigace na obrázku 4.3 vpravo. Kružnice je vykreslena se středovým bodem v místě poslední zachycené souřadnice. Značka udávající směr letu (resp. směru na cíl) je vykreslena pod úhlem, který

plyne z poslední a předposlední (resp. cílové) souřadnice. Nejkratší trasa mezi dvěma body na zeměkouli nevede po přímce, ale po oblouku. Výpočet tzv. ortodormické vzdálenosti je popsán na str. 38 v publikaci [5].

$$\theta = \arctan(\frac{cos(\phi_1)\sin(\phi_2) - \sin(\phi_1)\cos(\phi_2)\cos(\Delta\lambda)}{\sin(\Delta\lambda)\cos(\phi_2)}) \tag{4.1}$$

V rovnici 4.1 θ udává kurs ze souřadnice ϕ_1, λ_1 do souřadnice ϕ_2, λ_2 a $\Delta\lambda$ udává rozdíl složek zeměpisné délky obou souřadnic. Další přizpůsobená třída *AltitudeGraph* pouze vykresluje závislost nadmořské výšky na celkové vzdálenosti v průběhu letu. Vzhled je popsán výše na obrázku 4.2. Poslední nestandardní třídou je *WindGraph* vyobrazen na 4.4. Tato komponenta neobsahuje žádné složitější výpočty a stejně jako obě dvě předchozí k vykreslení používá rozhraní frameworku *Core Graphics*.



Obrázek 4.11: Diagram tříd skupiny View. Všechny třídy implementující grafické prvky na míru dědí své vlastnosti ze standarní třídy UIView, což jim umožňuje snadnou integraci do aplikace.

Všechny třídy spadající do kategorie *controller* jsou přímo určeny návrhovým vzorem *Model-View-Controller*. Jak je dobrými praktikami programovaní pro *iPhone*, každé obrazovce aplikace odpovídá právě jedna třída typu *controller*, která se stará o události spojené nejen s interakcí uživatele. *View* každé obrazovky — samotné grafické uživatelské rozhraní — není implementováno pomocí samostatných tříd, ale je definováno v nástroji *Interface builder* a uloženo v souberch typu *NIB*.

4.4.2 Obrazovky

Obrázek 4.12 představuje navrhnuté obrazovky, jejich návaznost a logiku přechodů mezi nimi. Základ tvoří lišta se třemi záložkami. První — nejvíce vlevo — odpovídá pilotnímu deníku, druhá — uprostřed — reprezentuje záznam a letovou navigaci a konečně třetí záložka zprostředkuje přizpůsobení aplikace. V obrazovkách skrývajících se pod prvními dvěma záložkami je v horní části umístěna ještě jedna lišta, která slouží k zanořovaní do (resp. vynořování z) hierarchie obrazovek. Tato navigační lišta obsahuje kromě ovládacích prvků pro práci s obrazovkami také titulek aktuálního zobrazeného obsahu. Nutno podotknout, že navigační lišta není výhradní obsluhou pro práci s okenní hierarchií. Neboli zanořování a vynořování obrazovek mohou způsobit i jiné prvky než jen ty umístěné na navigační liště. Struktura zanoření odpovídá datové struktuře *zásobník* (*last in, first out*). Na obrázku je u každého přechodu popsáno, jak se aktivuje. Pokud není konkrétně uvedeno jakou akcí se mezi obrazovkami přechází, např. *Výběr konkrétního letu*, jednoslovné označení přechodu odpovídá popisku tlačítka vyvolávajícího daný přechod, které se zpravidla nachází na navigační liště jako např. *Flights*.

Obrázek 4.12: Diagam návaznosti obrazovek a logiky přechodů. Ve spodní části každé obrazovky je zvýrazněna aktuální záložka. Konec běhu nastává po stisknutí tlačítka pro návrat.

- Obrazovka *Fligts* obsahuje přes celou plochu standardní tabulku s daty absolvovaných letů. Šipka v každém řádku tzv. *disclosure indicator* tabulky indikuje, že let obsahuje dostatek zaznamenaných souřadnic a je možné přejít na obravku s detaily popsanou v dalším bodu. Tabulka obsahuje posuvník pro případ, že by počet zaznamenaných letů přesáhl její velikost.
- *Flight detail* obsahující podrobnosti o vybraném letu je strukturovaná do podobné tabulky jako seznam provedených letů s tím rozdílem, že spolu související informace jsou shromažďovány do oddílů. Informace o místě přistání obsahuje buď textový název města nebo okresu, pokud se jej podaří programově získat, nebo GPS souřadnice daného místa. Dále stojí za zmínku poslední oddíl této tabulky, kde jsou řádky obsahující hesla *Map* a *Altitude profile* stejně jako v předchozím bodě opatřeny značkou *disclosure indicator*. Výběr těchto buněk způsobí další zanoření do obrazovek na obrázku 4.13 obsahujících trasu letu na interaktivní mapě a výškový profil letu. V horní navigační liště je vždy vlevo tlačítko s názvem obrazovky o úroveň výš umožňující vynoření.
- *Detail map* s vyznačenou trasou letu dovoluje měnit úroveň přiblížení a posun po mapě stejně jako v původní aplikaci *Maps*, která je součástí každé instalace *iOS* a na kterou je uživatel zvyklý. Způsoub zobrazení je tzv. hybridní, což znamená viditelnost terénu s popisky.
- *Altitude profile* obsahuje jedinou komponentu s grafem nadmořské výšky v závislosti na uražené vzdálenosti popsanou v podkapitole 4.2 o prvcích uživatelského rozhraní.

Obrázek 4.13: Obrazovky detailu letu *Detail map* a *Altitude profile*.

- Druhá obrazovka na nejvyšší úrovni *Navigation* slouží k ovládání záznamu letu a k nastavení cíle. Zobrazuje také přesné informace ze senzoru *A-GPS*. Cíl zadává uživatel pomocí vysouvací softwarové klávesnice do textových polí ve formátu GPS souřadnic. Pro maximalní informovanost pilota je vždy o poslení získané souřadnici vypsána horizontální a vertikální přestnost v metrech, stáří souřadnice v sekundách, zeměpisná šířka a délka, nadmořská výška, kurs pohybu, aktuální rychlost a čas. Uplně dole je tlačítko *Take off* (resp. *Land*) pro start (resp. ukončení) záznamu. Na navigační liště vpravo je tlačítko zanoření do obrazovky s mapou.

- Celou obrazovku *Map* pokrývá interaktivní mapa s navigační kružnicí popsanou výše. Z navigační lišty se vynořují dva zaoblené obdélníky. Levý poloprůhledný nese nejdůležitější informace o poslední obdržené souřadnici. Pravý neprůhledný obdélník s nápisem *Wind* má dvojí funkčnost. Po klepnití jednak slouží pro vytažení rolety s diagramem obsahujím vektory větru a jednak dovoluje další zanoření do obrazovky s tabulkou se standardními značkami. Jakou funkčnost obdélník zastává, záleží na uživalském přizpůsobení aplikace, které je popsáno níže v posledním bodě. Roleta s diagramem obsahuje záhlaví, kde jednotky rychlosti závisí také na přizpůsobení aplikace, a samotný graf, který je podle velikosti obsahu vybaven posuvníkem, viz obr. 4.12. Detailně je graf rozebrán výše v podkapitole 4.2. Zvětšení (resp. zmenšení) intervalu letových hladin funguje standardně, stejně jako např. u mapové komponenty tak, že uživatel poklepe (resp. klepne dvěma prsty) na oblast grafu.

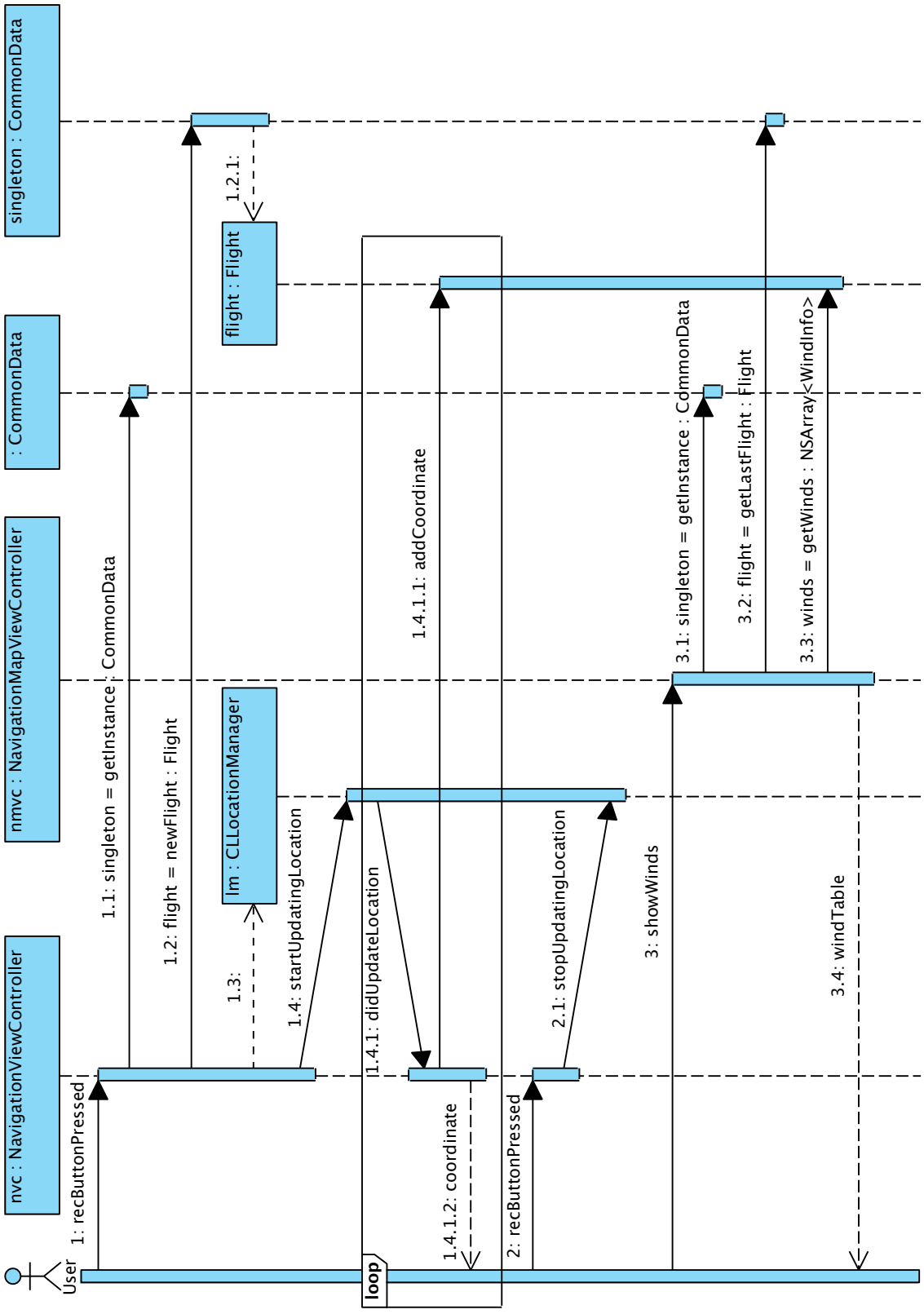
- *Winds* je jednoduchou tabulkou obsahující pro každou letovou hladinu standardní značku pro vektor větru a tutéž informaci textově, přesně ve stupních a v *m/s* nebo *km/h*. Volbu letové hladiny zprostředkuje kontextová nabídka předložená po stisknutí tlačitka *Levels* vpravo na navigační liště.

- Na obrazovce *Settings* jsou možnosti přizpůsobení aplikace. Mezi ně patří název balónu, způsob zobrazení větrné mapy a jednotky rychlosti.

Tímto je statická struktura aplikace popsána.

4.4.3 Dynamický popis vybrané sekvence

Na obrázku 4.14 je uveden dynamický diagram sekvence záznamu letu obsahující obdržení a zpracování nové souřadnice z *A-GPS* senzoru. Na něj navazuje požadavek uživatele na zobrazení větrné mapy. Diagram uvozuje první událost, což je stisk tlačítka *Take Off/Land* pro počátek nahrávání na obrazovce *Navigation*. Zprávu o této události obdrží controller dané obrazovky *NavigationViewController*, který kontaktuje model aplikace s požadavkem o vytvoření nové instance třídy *Flight*. Dále vytvoří a inicializuje instanci třídy *CLLocationManager*, která tvoří rozhraní pro přístup k datům z *A-GPS* senzoru. Po spuštění odběru nových souřadnic controller naplňuje trasu aktuálního letu a zároveň zobrazuje aktuální informace uživateli. Jakmile uživatel ukončí záznam, controller ukončí sběr nových GPS souřadnic. Druhá sekvence figuruje na obrazovce *Map* a je uvozena uživatelovým požadavkem na zobrazení aktuální větrné mapy. Odpovídající controller z modelu aplikace získá referenci aktuálního letu, který požádá o vygenerování větrné mapy pro daný interval letových hladin. Graf nebo tabulka je prezentován na základě uložených preferencí uživatele. Sekvence ostatních případů užití jsou buď analogické nebo triviální, aproto zde nebudou zdlouhavě rozebírány.



Obrázek 4.14: Diagram sekvence záznamu letu a zobrazení větrné mapy detailně popsaný v sekci 4.4.3.

Kapitola 5

Implementace

Implementační problémy se dají rozdělit do tří skupin. V první budou ty, které programování pro *iOS* vyžaduje a samo programátorovi podsouvá, jako jsou některé návrhové vzory, řešení struktury a přechodů mezi obrazovkami, práce s mapami atd. Druhou skupinu tvoří body aplikace, kde má vývojář volnější ruku a muže si vybrat z několika možných variant, z nichž některé jsou podporovány přímo od tvůrců systému, a jsou k nim dostupná programová rozhraní, anebo může zvolit naprosto nezávislou cestu a naprogramovat vlastní implementaci daného problému. Do této skupiny patří přístup k modelu aplikace, řešení persistence dat, uživatelské přizpůsobení aplikace a další. Třetí a poslední jsou ty oblasti, kde je tvůrce aplikace odkázán jen sám na sebe a musí si vystačit s vlastními silami. Jako příklad poslední skupiny poslouží prvky uživatelského rozhraní na míru uvedené v podkapitole 4.2. V některých případech existuje možnost použít knihovny třetích stran, ale to se konkrétně v této aplikaci ukázalo jako zbytečně robustní a zdlouhavé řešení. V této kapitole budou uvedeny nejkritičtější a nejzajímavější aspekty implementace, které bylo nutné pro zdárné splnění zadání zdolat. Mezi obecné zásady, které respektuje celá aplikace, patří Směrnice pro lidské uživatelské rozhraní dostupné na [12]. Důraz je kladen na konzistenci uživatelského rozhraní, což se projevuje v ovládání různých prvků takovým způsobem, který uživatel intuitivně očekává a na který je zvyklý z jiných aplikací. Tam, kde je to možné, jsou využity techniky *lazy loading* a recyklace objektů.

5.1 Implicitní a vybrané použité návrhové vzory

Jak už bylo několikrát zmíněno, struktura téměř každé komplexnější aplikace respektuje návrhový vzor *Model-View-Controller* teoreticky popsaný v podkapitole 3.3. U této aplikace tomu nebude jinak. Všechny úlohy jsou striktně rozděleny mezi odpovídajících třídy. Vzor *MVC* je aplikován u každé obrazovky bez výjimky. Detailně bude popsán jen jeden vybraný případ, jelikož všechny mají velice analogické rysy. Zvolme např. obrazovku *Navigation*, která se pro připomenutí nachází na obrázku 4.12 nahoře uprostřed. Její vzhled je definován pomocí aplikace *Interface Builder* v souboru *NavigationViewController.xib* a třída controlleru je definována v souboru *NavigationViewController.m*. Níže je uvedeno ve zkrácené formě rozhraní třídy.

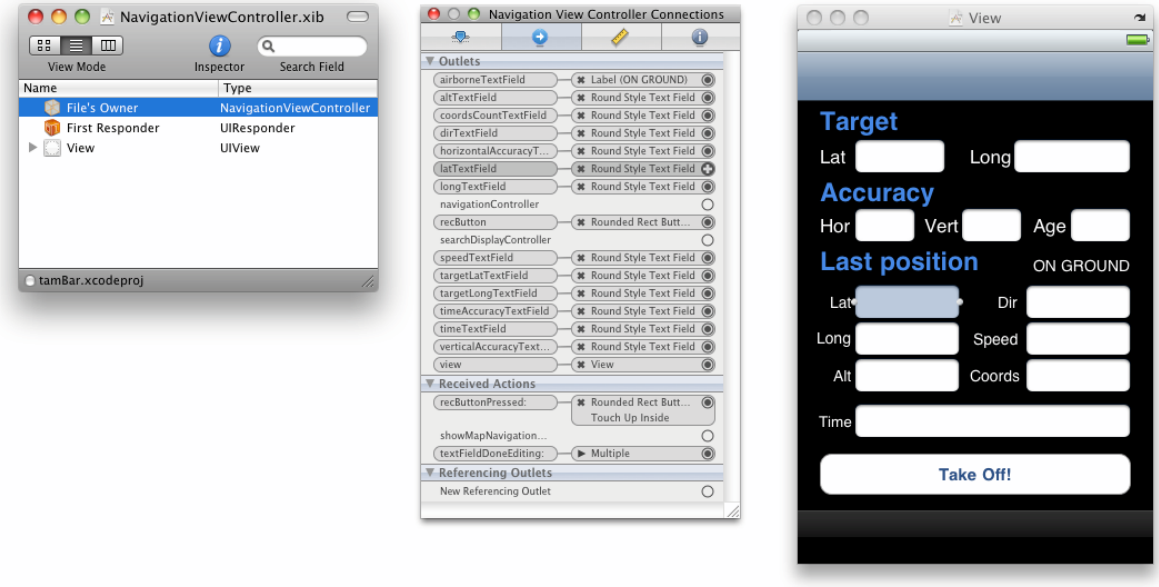
```
#import <UIKit/UIKit.h>
#import <CoreLocation/CoreLocation.h>
#import "CommonData.h"

@interface NavigationViewController : UIViewController <
    CLLocationManagerDelegate>{
    IBOutlet UITextField *latTextField;
    CLLocationManager *locationManager;

}

@property (nonatomic, retain) UITextField *latTextField;

- (IBAction)recButtonPressed:(id)sender;

- (void)updateLocation:(CLLocation *)newLocation;

@end
```

Propojení *view* a *controlleru* existuje ve dvou rovinách, jak je popsáno v 4.2. Zopakujme, že první slouží ke změně údajů (např. textových) na obrazovce. Ten je realizován pomocí konstrukce *IBOutlet* [1]. Pokud je reference definována jako v uvedeném příkladu *latTextField*, je možné programově číst a měnit parametry komponenty *UITextField*, která je připojena na uvedený *IBOutlet*. Toto propojení je realizováno v nástroji *Interface Builder* editací přílušného *NIB* souboru, jak je ilustrováno na obrázku 5.1. V prostředním okně jsou definované *outlety*, které stačí propojit tažením myši z kroužku vpravo na řádku u příslušného outletu do komponenty v pravém okně. Vlastníkem *NIB* souboru (*File's Owner*) je soubor, který jej otevírá a spouští, což je zpravidla odpovídající *controller*. V tomto případě je to *NavigationViewController*. Podobně jsou propojeny i požadované akce definované v hlavičkovém souboru jako funkce s návratovou hodnotou *IBAction* [2]. V uvedeném příkladě je spojena funkce *recButtonPressed* s akcí *touch up inside* na tlačítku *Take Off*. Typ akce je po natažení spojení vždy nabídnut dle typu grafické komponenty.

Obdobně jsou ustanovena všechna nezbytná propojení a tím vytvořeno obousměrná vazba *View-Controller*, podrobněji viz [4]. Přístup k modelu je realizován pomocí návrhového vzoru *Singleton*. Pouze pro tento účel je implementována třída *CommonData*, která obsahuje vektor všech zaznamenaných letů se statickou metodou *singleton* vracející její jedinou instanci v aplikaci. Jednotlivé lety jsou v kolekci *NSMutableArray* uspořádány podle data vzniku, což zaručuje, že případný aktuálně zaznamenávaný let je vždy k nalezení jako poslední prvek.

Návrhový vzor *delegate* je jedním ze základních stavebních kamenů *iOS*. Controllery se stávají delegáty, pokud implementují daný protokol (rozhraní). Aplikace implementuje několik typů delegátů. Jedním z nich je *UITableViewDelagate*, který implementuje mj. akce vyvolané vybráním řádku tabulky. Konkrétně je toto rozhraní součástí controlleru obrazovky *Flights*, která detekuje vybrání některého z letů a prezentuje jeho detail. Jiný protokol, který musí tento controller implementovat, nese název *UITableViewDataSource*. Metody, které jej naplňují, poskytují tabulce rozměry a data. V tomto případě metoda *numberOfRowsInSection* vrací počet zaznamenaných letů a *cellForRowAtIndexPath* buňky s daty jednotlivých letů. Různých delegátů se vyskytuje napříč celou aplikací několik. Nemá vý-

[1] *IBOutlet* – Interface Builder Outlet
[2] *IBAction* – Interface Builder Action



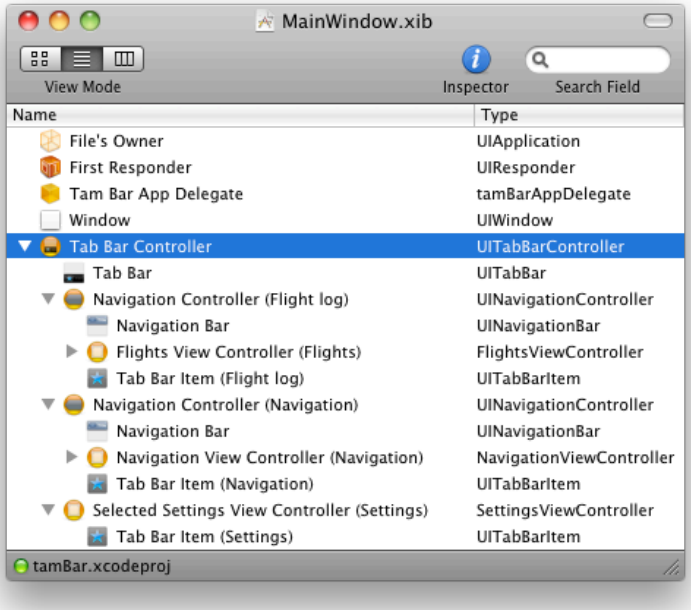
Obrázek 5.1: Použití nástroje *Interface Builder*. Šedým podbarvením je v prostředním okně vyznačen outlet, který je propojen s modře podbarveným textovým polem v pravém okně.

znam všechny rozebírat, jelikož jejich použití je principiálně stejné a vyžaduje nastudování programového rozhraní. Způsob použití jednoho z nich bude však nyní objasněno, jelikož je celá aplikační funkčnost postavena na jeho službách. Jedná se o objekt zmíněný ke konci kapitoly 4 *CLLocationManager* a protokol definující delegáta *CLLocationManagerDelegate*. Použití této třídy je však poměrně jednoduché. Její instanci postačuje kromě přiřazení delegáta specifikovat minimální přesnost, jakou mají mít obdržené informace o poloze, a zaslat zprávu *startUpdatingLocation*. Větší přesnost si bere daň v podobě větší spotřeby energie a tím snižuje výdrž baterie mobilního zařízení. Vybraný objekt delegát je prostřednictvím volání metody *didUpdateToLocation* informován o nové poloze zařízení, kterou dále zpracovává.

5.2 Kostra

V této podkapitole je popsána implementace základní kostry aplikace, kterou tvoří lišta se třemi záložkami u dna a navigační lišta pod horním okrajem obrazovky. K této konstrukci jsou využity dva speciální prvky, které jsou přímými potomky třídy *UIViewController*. Jsou jimi *UITabBarController*, který spravuje záložky, a *UINavigationController*, který obstarává zanořování a vynořování obrazovek v kontextu záložek *Flights* a *Navigation*. Tyto dva prvky porušují dvě pravidla modelu *MVC* a dříve stanovených zvyklostí. Jednak nejsou definovány v samostatném souboru a jednak se k nim neváže žádná obrazovka neboli *view*. Oba jsou specifikovány v *NIB* souboru hlavního okna aplikace *MainWindow.xib*, jehož obsah je znázorněn na obrázku 5.2. Nástroj *Interface Builder* tedy kromě vzhledu oken dovoluje definovat do jisté míry i aplikační strukturu.

V rámci prvku *Tab Bar Controller* se ke každé záložce váže prvek typu *controller*. U poslední záložky *Settings* je to přímo controller obrazovky s aplikačním nastavením. U dvou záložek nejvíce vlevo se jedná opět o generický prvek *Navigation Controller*. Každý



Obrázek 5.2: Stromová struktura hlavního okna aplikace odhaluje zanoření konkrétních controllerů v jednotlivých záložkách.

z těchto navigačních controllerů má specifikovaný kořenový přizpůsobený controller, ke kterému se již váže obrazovka, konkrétně *Flights* resp. *Navigation* a který je použit po přechodu na danou záložku. Zanořenější obrazovky jako *Flight Detail*, *Map* atd. jsou vždy vloženy na vrchol zásobníku obrazovek spojeného se svým kořenovým a potažmo navigačním controllerem.

5.3 Persistence

Nutnost nasbíraných dat překonat dobu běhu aplikace plyne přímo ze zadání. Otázkou bylo jak persistenci zajistit. Standardně jsou k dispozici čtyři varianty ukládání dat. Každý se hodí na jiný objem, formu a způsob práce s daty.

- Prvním z nich je tzv. *Property List*, který je serializovatelný a editovatelný z vývojového prostředí. Serializace zde probíhá podobně jako např. v jazyce *Java*. Objekt, který má být převeden na řetězec bytů, musí implementovat rozhraní protokolu *NSCoding*, které (narozdíl od *Serializable* v jazyce *Java*) není prázdné, ale obsahuje metody *encodeWithCoder* a *initWithCoder*, umožňujících archivaci a opačný proces. Problém této varianty je, že *Property List* nemůže obsahovat objekty libovolného typu, ale pouze několik vyvolených. Pro potřeby této práce se tedy příliš nehodí.
- Druhý způsob je strukturovaní modelu aplikace do jediného objektu a ten serializovat. Jelikož je tento objekt — atribut pole letů třídy *CommonData* — již v navrhnuté struktuře připraven, jeví se tato varianta jako poměrně přímočará. Nevýhodou může být pomalejší práce s archivem závisející na parsování uloženého XML stromu.

- Další možností je využití vestavěné *SQLite3* databáze, se kterou je možné pracovat rychleji než se serializovanými objekty. Její použití je však daleko komplikovanější a je tedy na zvážení, jestli dát přednost rychlosti před snadným použitím.
- Poslední varianta vychází z frameworku *Core Data* a jedná se o objektově relační mapování modelu a uložení opět ve *SQLite3* databázi. Toto je vzhledem k jednoduchosti modelu aplikace zbytečně robustní řešení, které by našlo využití spíše v rozsáhlých a komplexních aplikacích.

Více a podrobněji je možné se dozvědět o všech uvedených způsobech v kapitole 12 publikace [4]. Vzhledem k faktu, že efektivita persistence leží mimo těžiště tohoto projektu, rozhodl jsem se pro nejpřímočařejší řešení serializace kolekce provedených letů. Další usnadnění implementace persistence představuje uložení celé kolekce do jednoho souboru, což by v případě rozsáhlého a dlouhodobého používání aplikace jistě vedlo ke zpomalení načítání celého archívu do paměti. Varianta s *SQLite3* databází je jistě šetrnější ve smyslu velikosti uloženého souboru, to se ale vzhledem ke kapacitě zařízení jeví jako irelevantní.

5.4 Zobrazování map

Řešení zásadního problému vizualizace map je vcelku triviální. Framework *MapKit* poskytuje komponentu *MKMapView*, která implementuje zobrazení mapových podkladů *Google Maps* v mapovém, satelitním i hybridním módu, přibližovaní a posouvání pomocí tradičních gest. Přes tyto mapy lze s využitím konstrukce delegát zobrazovat trasy, navigační růžice a obecně jakékoliv podtřídy *UIView*. Nevýhodou použití této komponenty je nevyhnutelnost výhradního užívaní mapových podkladů společnosti *Google Inc.*

Obrázek 5.3: Přizpůsobení v aplikaci *Settings* standardně přístupné na domovské obrazovce systému.

5.5 Nastavení

Uživatelské přizpůsobení je nedílnou součástí každé aplikace. V tomto případě existují tři přizpůsobitelné položky. Typ balónu, což je vzhledem k fungování aplikace poměrně nevýznamná položka, jednotky rychlosti (na výběr je mezi *m/s* a *km/h*) a varianta zobrazení směru a síly větru v letových hladinách (k dispozici je forma grafu a tabulky se standardními značkami, viz obrázek 4.4 vpravo a uprostřed). Logické členění struktury aplikace vykazuje nastavení do samostatné obrazovky (na obrázku 4.12 označena názvem *Settings*) pod třetí záložkou dostupné z jakékoliv obrazovky. K implementaci byl použit mechanismus *Defaults*, který obsahuje balík nastavení a je persistentní. Nastavení přetrvají mezi běhy aplikace a navíc je možné je editovat přímo v aplikaci *Settings*, která slouží k přizpůsobení celého systému. Na obrázku 5.3 je náhled na nastavení aplikace v rámci celkového nastavení systému. Teoreticky by bylo možné měnit přizpůsobení pouze přes tento nastavovací balík z *Settings*, ale pro snadnější použití byl implementován přístup k těmto parametrům i uvnitř samotné aplikace.

Kapitola 6

Vyhodnocení

Následující kapitola vyhodnocuje výsledný produkt a jeho testování. Testování probíhalo na několika úrovních. První fáze se zabývá rozborem aplikace na úrovni vytížení systému z hlediska alokované paměti, doby spuštění, velikosti uložených dat atd. Tyto testy se vztahují k aplikaci běžící v simulátoru, aby bylo možné dosáhnout přesných měření. Druhá čast se věnuje rozboru a vyhodnocení experimentu s aplikací v terénu během balónového letu, což už se týká instalace na mobilním zařízení. Dále bude zmíněno, jaká je na trhu v tomto sektoru konkurence a jaké má ve srovnání s konkurencí aplikace výhody a nevýhody. Závěrem budou uváženy kroky, které by následovaly při pokračování vývoje aplikace.

6.1 Výkon a paměť

Neprve bude rozebrána velikost uložených dat. Průměrný balónový let trvá od jedné do dvou hodin. Empiricky zjištěný minimální interval mezi jednotlivými souřadnicemi je *10s*. Pokud jsou sbírány častěji, trasa je příliš podrobná, nepřináší to žádná pozitiva a pouze to zpomaluje běh celé aplikace. Položíme-li délku letu hodinu a půl, počet souřadnice je $5400s/50s = 1180$. Velikost uloženého XML souboru s letem, který obsahuje tento počet souřadnic, je přibližně *180Kb*. Tato velikost je způsobena také nešetrným pojmenováním klíčů atributů serializovaných objektů standardních tříd. Pokud by nebyl pro uložení souřadnice použit objekt třídy *CLLocation*, ale objekt se stejnými atributy, který by měl pouze jiné klíče pro pojmenování XML elementů, byla by velikost uloženého souboru zhruba o 30 % snížena.

Další z měřených parametrů byl čas nutný ke spuštění aplikace. Doba byla měřena pomocí konzolových výpisů vždy od počátku sezení do času kontrolního výpisu, který následoval načtení letů do paměti. Subjektivně byl start aplikace svižný do počtu přibližně 75 načítaných letů. Čas nutný pro spuštění se pohyboval pod 5 sekundami. Při více uložených datech v paměti už se start aplikace jevil zdlouhavě. Tento osobní pocit byl mírně potlačen nastavením spouštěcí obrazovky ve vzhledu spuštěné aplikace bez načtených letů. Spouštěcí obrazovka je obrázek ve formátu *png* o stejných rozměrech, jako má displej zařízení, který se vykreslí ihned po klepnutí na ikonu aplikace a je viditelný až do uplného spuštení. Nutno podotknout, že na tento aspekt nebyl kladen při implementaci přílišný důraz. Vyřešení problému je popsáno v podkapitole 6.3 o dalším vývoji aplikace.

Objem alokované paměti zabrané kostrou aplikace není nikterak významný. Roste však se sumou souřadnic všech letů v paměti, což se nejvíce projevuje při vykreslování dráhy letu do mapy, jelikož je pro tento účel trasa připravena do speciální struktury a dochází

k duplikaci souřadnic v paměti. Výše zmíněné zátěžové experimenty s výkonem aplikace byly prováděné s množstvím dat, které odpovídá několikaleté zátěži při běžném používání. Parametry mobilního zařízení tedy bohatě dostačují objemu dat odpovídajícímu předpokládanému využití.

6.2 Experiment v praxi

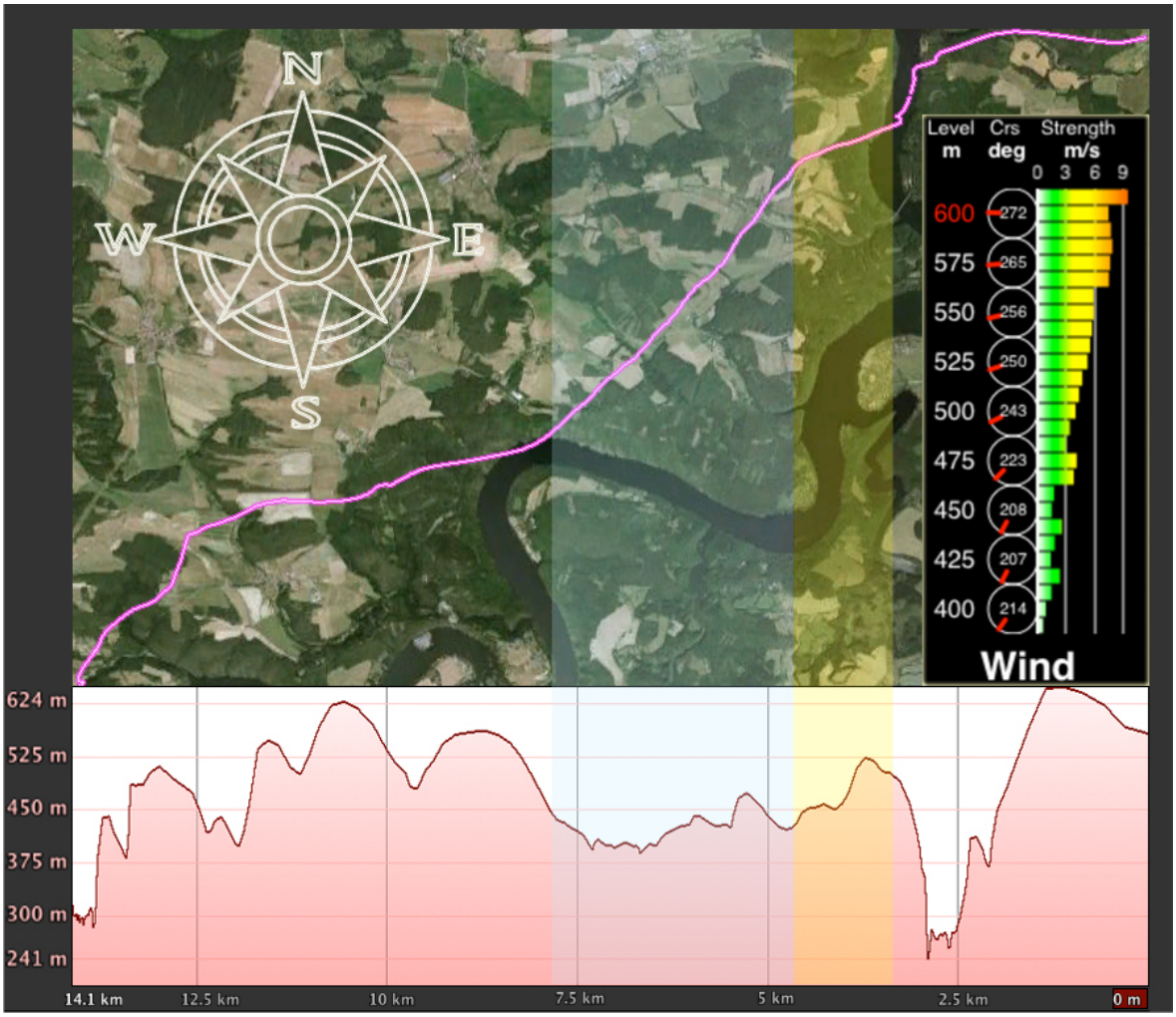
Testování probíhalo na zařízení iPhone 3GS s operačním systémem ve verzi 4.2.1. Celkem byly provedeny tři balónové lety s aplikací na palubě. Během prvních dvou letů byly odladěny některé drobné implementační nedostatky. V průběhu třetího letu již byla navigace plně využita k vedení balónu požadovaným směrem. Bohužel nebyl ani jeden z letů soutěžní. Nebylo požadováno letět na konkrétní cíl, a tak nebylo ověřeno, zda by aplikace dokázala provádět naprosto přesnou navigaci. Korelaci větrné mapy s opravdovou meteorologickou situací ukazuje následující snímek. Pro názornost bylo využito mapové komponenty a výškového profilu z programu Google Earth. Barevné zvýraznění a graf větrných vektorů nepocházejí z tohoto programu.

Obrázek 6.1 vznikl složením několika částí za účelem demonstrace funkčnosti aplikace. Mapová část a graf závislosti jsou, jak již bylo zmíněno, propůjčeny z aplikace Google Earth. Tabulka s letovými hladinami, směry větru a grafem síly větru jsou vyňaty z aplikace, která je předmětem této práce. Graf vznikl během letu zvýrazněného na mapě. Let měl jihozápadní tendenci, začínal tedy v pravém horním rohu mapy. Proložíme-li ve kterémkoliv bodě svislou přímku celým obrázkem, nadmořská výška v místě průsečíku přímky s dráhou letu odpovídá hodnotě grafu v místě průsečíku s touto přímkou, neboli horizontální průběh kopíruje vertikální. Aby graf koreloval s průběhem letu a s vypočítanou větrnou mapou, bylo nutné netradičně vést jeho průběh zprava doleva.

V obrázku jsou zakresleny dva svislé poloprůhledné pásy, které slouží ke zvýraznění přibližné polohy balónu v určité letové hladině. Uvažujme nejprve úsek, kdy balón prolétal nad řekou Vltavou, těsně před vstupem do žlutého pásu. Z výškového profilu lze odečíst nadmořskou výšku v této oblasti zhruba 280 m $n.m.$ a téměř jižní kurs letu. Tato oblast podává velice zkreslené informace, neboť se jedná o místo těsně nad hladinou řeky a v jejím údolí. Vektor větru je silně ovlivněn terénem a neodpovídá reálné meteorologické situaci. Tuto odchylku je velice problematické ošetřit a aplikační navigace se bude muset omezit na nadmořské výšky dostatečně vzdálené od terénu.

Před vstupem do žlutého pásu pilot nastoupal do hladiny nad 500 m $n.m.$ a nabral, jak lze odečíst z větrné mapy, kurs přibližné 250°, který udžel po celou dobu průletu žlutým pásem a nebyl příliš vychýlen jižním směrem, přestože klesl do hladiny 450 m $n.m$, kde tímto směrem vane. V této hladině je totiž rychlost větru nižší než hodnota 3 m/s. Nicméně po vstupu do modrého pásu se balón ustálil v této hladině (pod 450 m $n.m$ a získal jihozápadní kurs, přibližně 210°). Koncem modrého pásu balón začal stoupat a dostal se až do 550 $m.n.m$, kde podle větrné mapy opět začal směřovat více na západ, což odpovídá i trase na mapě.

V neposlední řadě je třeba se zamyslet nad sílou větru. V tomto případě platí přímá úměra nadmořské výšky a síly větru, což se na výškovém profilu projevuje strmějšími lokálními extrémy v nižších polohách a plošími extrémy v polohách vyšších. Lokální extrém odpovídá jedné dávce ohřátí vzduchu uvnitř balónového pláště. Pilot se ve vyšší letové hladině pohyboval rychleji a urazil na jednu dávku delší vzdálenost, což se ve výškovém grafu projeví jako plochá konvexní oblast.



Obrázek 6.1: Ukázka funkčnosti aplikace. Mapa a výškový profil dráhy pocházejí z aplikace Google Earth. Směrová růžice, diagram větru a barevné pásy jsou doplněny pro názornost a lepší pochopení korelace větrné situace s nasbíranými daty. Barevný pás značí let v jedné konkrétní letové hladině.

Tímto je popsána valná většina trasy letu pomocí získané větrné mapy, což lze považovat za důkaz, že směry a rychlosti větru v jednotlivých hladinách skutečně odpovídají meteorologické situaci a povětrnostním podmínkám v místě a okamžiku pořízení tohoto záznamu.

6.3 Srovnání s produkty na trhu a další vývoj

Tato podkapitola rozebírá konkurenční aplikace a produkty, které jsou na trhu aktuálně dostupné. Dále se zabývá, jakým směrem by se měl ubírat další vývoj aplikace, aby překonala všechny nedostatky, maximálně pokrývala požadavky cílového uživatele a zvýšila svoji konkurenceschopnost v tomto odvětví.

6.3.1 Existující produkty

Do konce roku 2010 nebyla na trhu s aplikacemi pro *iOS* obdoba této práce. Exitovaly pouze různé variace GPS loggerů více či méně přizpůsobených pro zaznamenávání balónových letů. Např. aplikace Hot Air [10] dokázala poměrně dobře zaznamenávat data o provedených letech. V prosinci roku 2010 nová verze této aplikace implemetovala tabulku se směry a silami větru pro různé letové hladiny a 11. května 2011 se v *App Store* objevila poslední verze, která navíc implementuje zobrazování libovolných oblastí na mapě definovaných importovaným KML souboru. Této aplikaci se však nedostává zásadní funkčnosti, kterou je volba intervalu letových hladin. Nemožnost upravit konstatní interval $500m$ nevyhovuje požadavkům balónových pilotů a lze ji považovat za podstatný nedostatek. Aplikace stojí 3,99 $ a dle dostupných informací je vyvíjena jednotlivcem.

Jinou alternativou k aplikaci vybudouvané během této práce jsou dedikované přístroje, které vycházejí z příručních GPS lokátorů. Jedním z nich je model *6040* společnosti *Flytec*. Mezi výhody přístroje označeného *Flytec 6040* patří například vestavěný barometr pro přesnější určení aktuální výšky a vyměnitelné baterie. Nevýhodami jsou černo-bílý displej s rozměry 320 na 240 bodů a především cena. Výrobce na svých stránkách, viz [9], uvádí částku 1.700 $. *Apple iPhone 4 16GB* je v ČR k dostání bez závazku obvykle za sumu přibližně 16.000 Kč, viz [18] a [20], což je (podle aktulního kurzu ze dne 13.5.2011) o něco více než polovina ceny *Flytec 6040*. Cena za aplikaci se předpokládá do 10 $, což je vzhledem k uvedeným sumám zanedbatelná částka.

6.3.2 Pokračování ve vývoji

Další kroky vývoje je možné rozdělit do dvou skupin. V první skupině jsou kroky implementující zadání v plné míře rozsahu, aby aplikace dosáhla své komplexní funkčnosti. Rozsah práce vyžadoval výběr konkrétní oblasti aplikace (více v kapitole 4), která má zásadní význam a kterou bylo třeba plně implementovat. Je třeba dokončit a rozšířit funkci pilotního deníku, který by ve výsledku obsahoval daleko více informací zadávaných pilotem, jako např. členy posádky, spotřebované palivo a letovou hmotnost, kterou by mohl doplnit kalkulátor určující schopnost vzletu při daném nákladu. Navigační část doplnit o zadávání cíle letu na mapě a změnu cíle během letu. Nedostatek současné implementace by při velikém počtu letů (přesahující testované množství v podkapitole 6.1) představoval způsob zajištění persistence dat. Všechny lety jsou uloženy do jednoho souboru a náhrány do paměti vždy současně. Jeden z dalších kroků by zajišťoval ukládání letů do samostatných souborů a čtení až v případě potřeby.

Druhá skupina kroků dalšího vývoje se částečně prolíná s tou první, neboť hranice zadané funkčnosti a implementovaných doplňků nebyla detailně specifikována. Jde tedy o vývoj nad rámec zadání, který by aplikaci ve všech směrech postavila nad konkurenci. Jednotlivé nápady a položky dalšího rozšiření jsou pro přehlednost uvedeny strukturovaně.

- Zlepšující se dostupnost služeb poskytujících předpověď meteorologické situace umožní doplnit větrnou mapu aspoň přibližnými údaji ještě dříve, než budou získány přesné údaje ze senzorů mobilního zařízení. To by mohlo urychlit a zpřesnit navigaci zkrácením času potřebného pro úvodní mapování povětrnostních podmínek.

- Export nasbíraných dat v podobě KML trasy nebo mapy s vyznačenou trasou letu distrubuovanou přes email nebo sociální síť patří mezi funkce, které by jistě zpříjemnily uživatelskou zkušenost s aplikací. Naopak užitečný může být import oblastí a tras

opět v KML formě vyznačující letiště, elektrárny, vedení vysokého napětí apod., který by pilotovi jistě usnadnil orientaci v terénu. Tato funkčnost se do jisté míry vyskutuje již u konkurenční aplikace popsané výše.

- Integrace komunikace s exerními senzory pomocí technologie bluetooh by zajisté pilotům přinesla užitečné informace při vyhodnocování nejen aktuální situace, ale zpětně i celých letů. Nabízejí se např. barometr, teploměr (na plášti nebo volně v prostoru), letecký rychloměr aj. Tyto přistroje komunikují směrem k mobilnímu zařízení. Poněkud utopická myšlenka uvažuje zařízení komunikující opačným směrem, které by bylo napojené na hořák balónu a lano větracího otvoru. Pak by aplikace byla schopná sama balón ovládat a teoreticky by zastávala funkci autopilota. Tato úváha však naráží na podmínky společnosti *Apple* kontrolující vystavení aplikace v *App Store*. Dokument *App Store Guidelines* dostupný pro členy placeného vývojářského programu na [6] zakazuje aplikacím autonomní řízení dopravních prostředků.

- Jistě užitečnou službou by bylo propojení dvou instancí této aplikace, kdy jedna by byla v balónu a fungovala by za podmínek uvažovaných v celé práci a aplikace spuštěná ve druhém zařízení umístěném v doprovodném vozidle by zobrazovala data z první instance. Rádiová komunikace je mnohdy neobratná a zdlouhavá. Posádka doprovodného vozidla by věděla aktuální pozici i meteorologickou situaci v oblasti balónu a snadněji by předpovídala místo přistání.

- Mezi drobnější úpravy algoritmického charakteru, které zpříjemní a usnadní použití aplikace, lze zařadit inteligentní filtraci obdržených souřadnic dle frekvence nebo odchylky či schopnost rozpoznat vzlet a přistání.

- Aby bylo možné aplikaci distrubuovat oficiálními kanály, bylo by nutné uvést ji do stavu, který vyhovuje podmínkám přijetí do *App Store*. Toto rozšíření je spíše formálnějšího charakteru, avšak není zanedbatelné. Jisté další drobnější úpravy, které by byly před uvedením na trh žádoucí, by povolily instalaci na zařízení *iPad*.

Rozšíření se nabízí celá řada, nejedná se ovšem o krátkodobou záležitost. Prostor pro rozvoj takové aplikace je široký a k dosažení komplexního a kvalitního výsledku by bylo zapotřebí většího množství zdrojů než v případě této práce.

Kapitola 7

Závěr

Cílem práce bylo vyvinout aplikace, která usnadňuje práci pilota horkovzdušného balónu. Dílčím výstupem první části je soubor informací potřebný pro návrh specifikované aplikace. Jednak se podařilo shromáždit informace o balónovém létaní a funkčních požadavcích na zadanou aplikaci a jednak informace nutné pro vývoj na platformě iOS dle aktuálních trendů. Podstatým aspektem pilotování balónu je princip jeho ovládání spočívající v nastavení letové hladiny, ve které má vítr požadovaný směr. Problémem je tedy správná volba letové hladiny. Ve spolupráci s cílovým uživatelem byla všechna nasbíraná fakta uvedena do podoby, která vyhovuje podmínkám použití a nárokům balónových pilotů. K tomu bylo využito primárního sběru dat v podobě dotazníkového šetření.

Byl vytvořen návrh aplikace splňující veškeré náležitosti plynoucí ze shromážděných dat. Vzhledem k potenciálnímu rozsahu aplikace byla stanoveno těžiště celé práce, kterému byla věnována větší pozornost než podpůrným oblastem. Hlavní úsilí bylo tedy směřováno na získání a zpracování aktuální meteorologické situace. Větrná mapa skládající se ze směru a síly větru pro jednotlivé letové hladiny je nejužitečnějším prvkem nápomocným v rozhodování pilota. Podstatným aspektem byla její grafická reprezentace a způsob interakce s ní. Implementované řešení na malém prostoru displeje poskytuje dostatek přehledných informací a představuje užitečný nástroj využitelný v praxi. Mezi další funkce aplikace se řadí vedení pilotního deníku, zpětné prohlížení detailů provedených letů včetně trasy a výškového profilu a zobrazování aktuálních informací ze senzoru A-GPS.

Aplikace byla v praxi otestována během série třech reálných balónových letů. Detaily jednoho z nich byly analyzovány a prokázaly korespondeci vygenerované větrné mapy s tehdejší meteorologickou situací. Tento experiment ověřil správnost a použitelnost výsledku práce.

Literatura

[1] Ahrens, C. D.: *Meteorology Today: An Introduction to Weather, Climate, and the Environment*. Thomson Brooks/Cole, 2007, iSBN 0-495-01162-2.

[2] Akerstedt, H.: CIA LIST OF NOTABLE PERFORMANCES and ACHIEVEMENTS [online]. `http://www.ballong.org/notable/`, 2004-04-01 [cit. 2011-01-02].

[3] Gamma, E.: *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 2000, iSBN 0201633612.

[4] Mark D., Nutting J., LaMarche J.: *Beginning iPhone 4 development: Exploring the iOS SDK*. Apress, 2011, iSBN 978-1-4302-3024-3.

[5] Stavovčík, I. B.: *Obecná navigace (061 00)*. AKADEMICKÉ NAKLADATELSTVÍ CERM, s r.o., 2008, iSBN 978-80-7204-576-1.

[6] WWW stránky: App Store Review Guidelines. `http://developer.apple.com/appstore/guidelines.html`.

[7] WWW stránky: Apple Inc. `http://www.apple.com/`.

[8] WWW stránky: FAI Ballooning commission - CIA. `http://www.fai.org/ballooning/`.

[9] WWW stránky: Flytec. `http://www.flytec.com/Products/Variometers/6040.htm`.

[10] WWW stránky: Hot Air - iPhone app for Balloon Pilots. `http://www.gano.name/shawn/hotair/`.

[11] WWW stránky: Hot Air Balloon Design. `http://www.hotairballoons.com/hotairballoon-design.asp`.

[12] WWW stránky: iOS Human Interface Guidelines. `http://developer.apple.com/library/ios/#documentation/userexperience/conceptual/mobilehig/Introduction/Introduction.html`.

[13] WWW stránky: iOS Reference Library. `http://developer.apple.com/library/ios/`.

[14] WWW stránky: iPhone uživatelská příručka. `http://manuals.info.apple.com/cs_CZ/iphone_uzivatelska_prirucka.pdf/`.

[15] WWW stránky: Letecká informační služba. `http://lis.rlp.cz/`.

[16] WWW stránky: NOAA - Air Resource Laboratory. http://ready.arl.noaa.gov/.

[17] WWW stránky: The Objective-C Programming Language. http://developer.apple.com/library/mac/documentation/Cocoa/Conceptual/ObjectiveC/.

[18] WWW stránky: Telefónica O2 Czech Republic, a.s. http://www.o2.cz.

[19] WWW stránky: vimov - iSimulate. http://www.vimov.com/isimulate/.

[20] WWW stránky: Vodafone Czech Republic a.s. http://www.vodafone.cz.

[21] WWW stránky: World Meteorological Organization. http://www.wmo.int/.

Příloha A

Dotazník pro balónové piloty

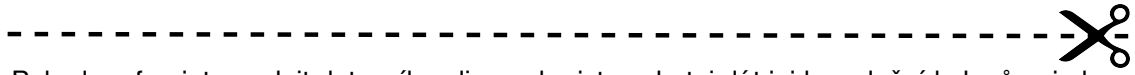
Dotazník obsahuje 15 otázek. Prosím, čtěte pár řádků na úvod...

Předmětem mé diplomové práce je vývoj aplikace s názvem **Asistent pilota balónu pro iPhone**. Aplikace je určena pro piloty horkovzdušných balónů, stojí na mobilní platformě iPhone a obsahuje dvě částí. První část zastává funkci elektronického pilotního deníku (logbook). Druhá část slouží k navigaci, zjišťuje aktuální meteosituaci - povětrnostní podmínky (windgram) a pomáhá navigovat pilota na zadaný cíl.

Následující otázky mají za cíl zjistit potřeby a požadavky balónových pilotů na takovou aplikaci. Zejména jakým způsobem informace získávat a zobrazovat s ohledem na podmínky balónového létání. Tento dotazník slouži pouze jako opora pro moji práci a napomáhá její využitelnosti v praxi. Uvedené údaje budou zveřejněny v technické zprávě pouze jako agregovaný výsledek a to anonymě.

Děkuji za spolupráci!

Ondřej Fabián,
VUT v Brně, Fakulta Informačních technologií,
xfabia00@stud.fit.vutbr.cz



Pokud preferujete vyplnit dotazník online nebo jste ochotni dát jej k vyplnění kolegům, je k dispozici na adrese:

http://www.stud.fit.vutbr.cz/~xfabia00/dotaznik/

nebo

http://1url.cz/wmu

Obecné

Tyto otázky nejsou povinné. Vyplňte je pokud máte zájem dozvědět se výsledky dotazníku, případně spolupracovat při vývoji/testování aplikace.

1. Jméno a příjmení

2. Email

3. Pohlaví ☐ muž ☐ žena

4. Mám zájem
   (zakřížkujte jednu nebo obě varianty)

   ☐ dozvědět se výsledek dotazníku ☐ podílet se na testování aplikace

5. Vlastním iPhone
   (zakřížkujte jednu variantu)

   ☐ ano ☐ ne

Profesní

Tyto otázky jsou povinné. Týkají se Vašich pilotních zájmů a zkušeností.

6. Proč létáte?
   (zakřížkujte převažující variantu)

   ☐ rekreační důvody
   ☐ obchodní důvody
   ☐ soutěžní/závodní důvody
   ☐ jiné

7. Jak dlouho létáte? (v letech)

8. Kolika balónových závodů jste se zůčastnil?
   (zakřížkujte jednu variantu)

   ☐ 0-4 ☐ 5-9 ☐ 10-14 ☐ 15-19 ☐ 20 a více

Logbook

Tyto otázky jsou povinné. Týkají se části aplikace, která poskytuje funkce elektronického pilotního deníku.

9. Které další informace o svém letu zaznamenáváte?
   (kromě obvyklých: čas a místo startu a přistání, doba letu, posádka)

10. Které informace o provedeném letu vás zajímají?
    (kromě obvyklých: čas a místo startu a přistání, posádka; zakřížkujte více variant; případně doplňte další)

- ☐ celkový čas letu
- ☐ celková vzdálenost
- ☐ maximální rychlost
- ☐ průměrná rychlost
- ☐ maximální vertikální rychlost
- ☐ průměrná vertikální rychlost
- ☐ maximální výška
- ☐ minimální výška
- ☐ zobrazení trasy letu na mapě
- ☐ další - napište vpravo

11. Jak rád/a zadávat cíl letu?
    (zakřížkujte jednu nebo více variant)

☐ GPS souřadnice ☐ na mapě

☐ jinak

Navigace

Tyto otázky jsou povinné. Týkají se části aplikace, která poskytuje navigaci během letu. Většina otázek nabízí 2-3 varianty s obrázkem a další 1-2 verianty bez obrázku.

12. Jaký **interval letových hladin** pro zobrazení směru a síly větru je pro vás dostačující
    (zakřížkujte jednu variantu)

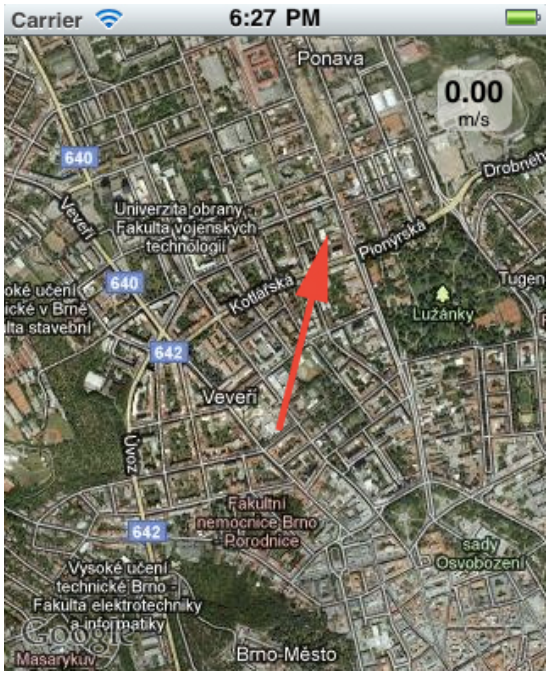
☐ 20m ☐ 50m ☐ 100m ☐ 150m ☐ 200m ☐ 250m

☐ jiný

13. Jaký způsob zobrazení **aktuální pozice** a **směru letu** považujete za nejpraktičtější?
(zakřížkujte jednu nebo dvě z následujících 5 variant)

☐ růžice

☐ kružnice s šipkou



☐ šipka

☐ tečka pro pozici a textový údaj pro směr (např. ve stupních)

☐ jiný

14. Jaký způsob zobrazení **směru na cíl** považujete za nejpraktičtější??
(zakřížkujte jednu nebo dvě varianty)

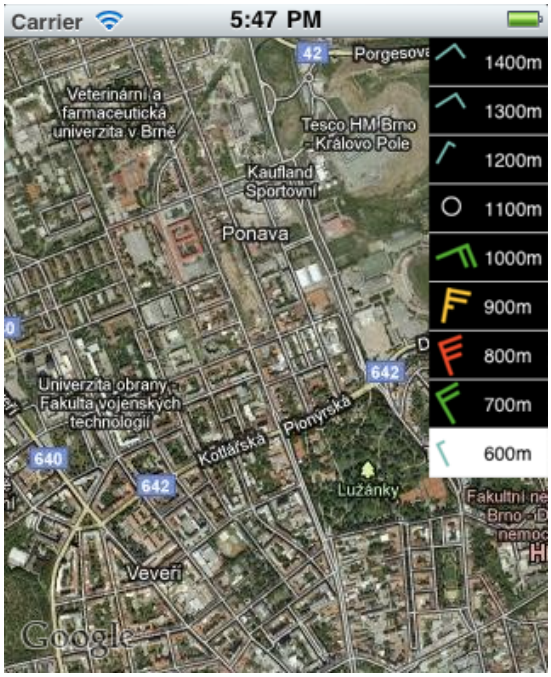
☐ prvek na okraji mapy

☐ prvek ve středu (trojúhleník)

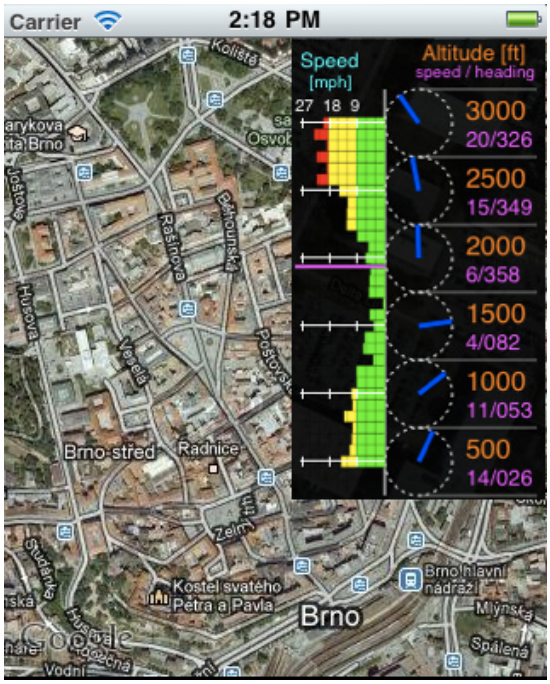
☐ textový údaj (např. ve stupních)

☐ jiný

15. Jaký způsob zobrazení **síly a směru větru** v jednotlivých letových hladinách považujete za nejpraktičtější?
(zakřížkujte jednu nebo dvě ze 4 variant)



☐ standardní značky



☐ diagram a směry

☐ šipky v prostoru

☐ jiný

Příloha B

Obsah CD

- Aplikace/ — složka obsahující projekt vývojového prostředí XCode.
- Data/ — složka obsahující testovací data.
- Technicka zprava/ — složka obsahující zdrojové texty technické zprávy v jazyku LaTeX.
- plakat.pdf
- readme.txt — návod ke spuštěni projektu a připojení testovacích dat.
- technicka_zprava.pdf – vysázena verze technické zprávy.
- zadani.pdf — zadání práce.