VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

INTELIGENTNÍ VYZVÁNĚNÍ PRO SYMBIAN S60

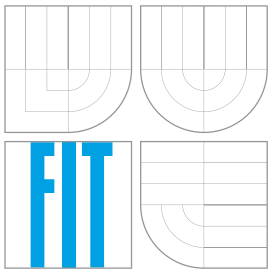
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. JAKUB KADLAS

BRNO 2010

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

INTELIGENTNÍ VYZVÁNĚNÍ PRO SYMBIAN S60

INTELLIGENT RINGTONES IN SYMBIAN S60

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. JAKUB KADLAS

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. JIŘÍ KOUTNÝ

BRNO 2010

Abstrakt

Symbian OS je v současné době jeden z nejrozšířenějších operačních systémů chytrých telefonů. Mezi jeho nejdůležitější zástupce patří platforma S60, kterou se zabývá tato práce. Část práce je zaměřena na teoretický přehled systému Symbian a popis integrovaných senzorů v chytrých telefonech. Druhá část je věnována realizaci ukázkové aplikace pro inteligentní vyzvánění. Ta umožňuje přizpůsobit vyzvánění telefonu okolnímu prostředí podle informací získaných ze senzorů. Aplikace je naimplementována v jazyce C++, s úpravami pro Symbian OS, ve vývojovém prostředí Carbide.c++. Využito bylo SDK verze S60 3rd Edition Feature Pack 1.

Abstract

Symbian OS is one of the most popular operating system for smartphones. The most favorite version of Symbian OS is the S60 platform, which is the aim of this thesis. Part of this thesis is focused on theoretical overview of Symbian OS, and a description of integraded sensors in smartphones. The second part is devoted to the implementation of sample application for inteligent ringtones. This application allows smartphone to customize phone ringing according to information obtained from sensors. Application was implemented in C++ language with Symbian-specific modifications. The development environment used was Carbide.c++ with SDK S60 3rd Edition Feature Pack 1.

Klíčová slova

Symbian OS, platforma S60, smartphone, Nokia, Carbide.c++, inteligentní vyzvánění, senzor, vyzvánění, příchozí hovor.

Keywords

Symbian OS, S60 platform, Nokia, Carbide.c++, inteligent ringing, sensor, ringing, incoming call.

Citace

Jakub Kadlas: Inteligentní vyzvánění pro Symbian S60, diplomová práce, Brno, FIT VUT v Brně, 2010

Inteligentní vyzvánění pro Symbian S60

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením pana Ing. Jiřího Koutného

.....
Jakub Kadlas
24. května 2010

Poděkování

Děkuji Ing. Petru Chmelařovi a Ing. Jiřímu Koutnému za jejich čas, obětavost, připomínky, cenné rady a informace, které mi při vypracování této diplomové práce poskytli.

Obsah

Kapitola 1

Úvod

1.1 Slovo úvodem

Mobilní komunikace se v posledních letech dostala do popředí technického dění. Výkony těchto mobilních zařízení se stále zvyšují. Ze zařízení dříve používaných pouze na telefonování se stávají náhrady přenosných počítačů. Samozřejmostí je připojení k internetu, pořizování audio i video záznamů, čtení elektronických dokumentů, GPS navigace a také přehrávání multimédií. Používání těchto zmíněných funkcí by se však neobešlo bez příslušného programového vybavení.

Díky velké rozšířenosti mobilní komunikace, zejména využitím chytrých telefonů, se otevřel nový segment pro vývoj aplikací. Tento trh není tak nasycen, jako u plnohodnotných počítačů, a z tohoto důvodu se dostává do pozornosti vývojářů aplikací.

Jako chytrý telefon (tzv. Smartphone) se většinou označují mobilní zařízení, které jsou vybaveny vlastním operačním systémem (dále také *OS*) a sadou aplikací umožňující tento mobilní telefon ovládat a spravovat. Smartphony nahradili dříve velice oblíbené PDA zařízení tím, že spojili jejich vlastnosti s vlastnostmi mobilních telefonů.

1.2 Cíl diplomové práce

Cílem této diplomové práce je vytvoření aplikace, která bude reagovat na okolní prostředí při příchozích telefonních hovorech a podle okolních vlivů měnit způsob vyzvánění. Aby šlo takovouto aplikaci vytvořit, je nutné znát cílovou platformu a její vlastnosti. Stejně tak je důležité znát chování jednotlivých senzorů a způsob získávání jejich dat. Veškeré informace potřebné k odpovědi na tyto otázky jsou rozepsány v následujících kapitolách.

1.3 Členění kapitol

Zde je uvedeno rozčlenění práce na jednotlivé kapitoly a jejich stručný popis. V první části je uveden teoretický rozbor problematiky a technologií, v druhé je popsána realizace tvorby programu.

V kapitole **Mobilní operační systémy** je popsáno rozdělení a historie současných nejpoužívanějších mobilních operačních systémů, jejich zběžná charakteristika, výhody a nevýhody. Používané programovací jazyky v těchto systémech.

Kapitola **Události v mobilních telefonech** popisuje práci s daty a událostmi v mobilních zařízeních, jako jsou reakce na zprávy, volání, atd. Speciálně se zaměřuje na komunikaci

s vestavěnými moduly a charakterizuje komunikaci mezi nimi a mobilním operačním systémem. Také je zde uvedena analýza akceleračního čidla, světelného čidla, fotoaparátu, záznamníku zvuku a GPS modulu.

V kapitole **Přístupy k analýze dat** je popsána analýza dat získaných z jednotlivých senzorů, volba metody k analýze a předpokládané výstupy. Popis metod získávání znalostí. Také popisuje klasifikaci dat získaných analýzou senzorových výstupů.

Následující kapitola **Návrh aplikace** přibližuje návrh a analýzu celkové aplikace pomocí jazyka UML.

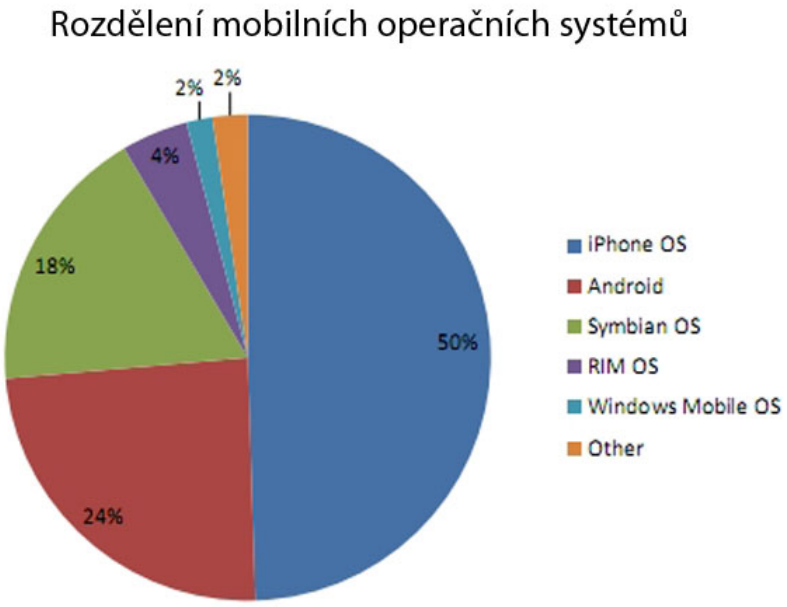
Kapitola **Implementace aplikace** se zaměřuje na výběr technologie, popis implementace, využitých nástrojů a řešení implementačních problémů. V této kapitole je také uveden popis aplikace, její chování, analýza nároků aplikace na systém a analýzy známých problémů při práci s aplikací.

V závěrečné kapitole se zhodnocují výsledky práce a možná rozšíření aplikace do budoucnosti.

Kapitola 2

Mobilní operační systémy

Nejčastějšími operačními systémy pro mobilní telefony jsou v současné době Symbian OS, Windows Mobile, Android a PalmOS [37]. Rozdělení podle počtu uživatelů je patrné z obrázku 2.1. Toto rozdělení se v posledních letech velice rychle mění. Hlavní vliv na pohyby v grafu má cenová politika firem a jejich úspěšné nebo neúspěšné snažení o optimalizaci operačních systémů.



Obrázek 2.1: Rozdělení operačních systémů [40].

2.1 Rozdělení mobilních operačních systémů

Nejpoužívanější mobilní operační systémy, které pokrývají většinu trhu s přenosnými zařízeními, jsou popsány níže. Informace byly čerpány z [37].

2.1.1 Symbian OS

Symbian OS je proprietární operační systém. Doplňují ho knihovny, grafické uživatelské rozhraní a referenční implementace nástrojů, které vytvořila firma Symbian Ltd. Symbian OS je následovníkem systému EPOC používaného v kapesních počítačích Psion a běží výhradně na procesorech ARM[1]. V současné době se jedná o operační systém reálného času. Poslední stabilní verze je Symbian[3].

2.1.2 Windows Mobile

Windows Mobile je operační systém spojený se základními aplikacemi pro mobilní zařízení založený na Win32 API z Microsoft Windows. Je navržen obdobně jako stolní verze systému Windows. Poslední stabilní verze má označení Windows CE 6.5.3.

2.1.3 Android

Android je na Linuxu založená softwarová platforma přednostně určená pro mobilní zařízení vyvinutá společností Google, která následně celou platformu i se zdrojovými kódy předala sdružení firem Open Handset Alliance, jejímž je také členem. SDK Android umožňuje vývojářům psát aplikace v jazyce Java s využitím knihoven vyvinutých společností Google. Tento OS má poslední stabilní verzi 2.1.

2.1.4 PalmOS

PalmOS je operační systém s grafickým rozhraním a intuitivním ovládáním, který je určený pro PDA a komunikátory. Jeho výhodou jsou nízké nároky na výkon a paměť, jelikož v jednom okamžiku může běžet pouze jedna aplikace. Při přepínání mezi aplikacemi si program uloží poslední stav, a po návratu na něj se aplikace spustí znova. Od verze 5, která je poslední stabilní verzí, umožňuje PalmOS spustit jednu až dvě aplikace jako rezidentní.

2.1.5 iPhone OS

iPhone OS je mobilní operační systém vyvíjen společností Apple Inc. Je to výchozí operační systém pro přístroje iPhone, iPod Touch a iPad. iPhone OS je odvozen od operačního systému Mac OS X, který patří do skupiny Unix operačních systémů. Je složen ze čtyř vrstev: jádra systému, základních služeb, multimediální vrstvy a vrstvy uživatelského rozhraní. Jeho primární způsob ovládání je pomocí dotykové obrazovky s minimem tlačítek. Tento operační systém zabírá v paměti přístroje přibližně 500MB.

První verze iPhone OS byla vydána v roce 2007. Původně nebylo povoleno instalování aplikací třetích stran. Apple Inc. od tohoto přístupu po nátlaku upustil a první vývojové prostředí pro iPhone OS bylo vydáno v roce 2008. Veškeré aplikace je však nutné instalovat přes App Store. Jedná se o službu firmy Apple, která dovoluje instalování aplikaci z iTunes Store[2]. Operační systém iPhone OS nedovoluje, v současné stabilní verzi 3.2, plnohodnotný multitasking, což je jeho současná největší nevýhoda [47].

[1] 32bitová mikroprocesorová architektura typu RISC.

[2] online softwarový obchod firmy Apple

2.1.6 Maemo

Maemo je svobodná vývojová platforma pro kapesní počítače a jiná mobilní zařízení postavená na bázi Debian GNU/Linuxu. Platforma je vyvíjená firmou Nokia a skládá se z operačního systému Maemo a Maemo SDK. Jeho hlavní výhodou jsou komponenty, které jsou založené na principech open source.

Poslední verze uživatelského prostředí Maemo 5 je založené na dotykovém ovládání. Má pevné pouze základní menu. Veškerý další obsah je na volbě uživatele. K dispozici jsou čtyři obrazovky pro naplnění zástupci jednotlivých instalovaných programů, widgetů, výstřižků internetových stránek, RSS čteček, aj. Uživatel tím dostává naprostou volnost v přizpůsobení přístroje [46].

2.1.7 Openmoko

Openmoko je projekt, který si klade za cíl vytvořit GSM smartphone platformu v duchu svobodného software. Je postaven na Linuxu a používá opkg balíčkovací systém, založený na ipkg[3]. Platforma umožňuje modifikování operačního systému i programů koncovým uživatelem [48].

2.2 Historický vývoj Symbian OS

Kapitola je zaměřená na stručné přiblížení vývoje mobilního operačního systému Symbian.

2.2.1 Symbian OS

Operační systém, v současné době znám jako Symbian, začíná svou historii vznikem společnosti Psion v roce 1980. Tou dobou firma vyvíjela software pro počítače ZX81 a ZX Spectrum.

Od roku 1984 začala společnost vyvíjet kapesní počítače Psion, které i přes značně omezené vlastnosti byly ve své době pokrokovým produktem. První verze neobsahovaly operační systém a byly podobné kalkulátoru. Následné produkty už obsahovaly operační systém EPOC s možností tvorby vlastních aplikací. V roce 1997 byla ve spolupráci s firmami Nokia, Ericsson a Motorola založena společnost Symbian Ltd. Operační systém EPOC byl přejmenován na Symbian OS a stal se základním kamenem nových chytrých telefonů.

Do roku 2000 se počet spolumajitelů společnosti rozšířil o firmy Panasonic, Sony a Sanyo. Konzorcium těchto firem vydalo první mobilní telefon s plnohodnotným operačním systémem Symbian v roce 2000. Nejednalo se ovšem o otevřenou platformu, nešlo tudíž instalovat vlastní aplikace. Následovalo období rozmachu a rychlého vývoje pokročilejších a modernějších verzí systému [42], [3].

- První telefon s otevřenou platformou Symbian OS v6.0, Nokia 9210, byl představen v červnu 2001, s implementovanou podporou Bluetooth, programovacího jazyka Perl a jazyka Java. Nejednalo se ovšem o Java ME standard, ale o speciální pJava a JavaPhone, určené pro mobilní telefony. Od verze 6.1 již můžeme mluvit o operačním systému Symbian S60. První verze 0.9 byla použita v mobilním telefonu Nokia 7650. Následující verze 1.2, známá jako S60 1st Edition znamenala průlom v multimediálních prvcích v mobilních telefonech. Díky mobilním telefonům Nokia N-Gage

[3] balíčkovací systém pro instalaci software z repositářů

se Symbian stal součástí života mladší generace a tím středem zájmu programátorských firem, které vyvíjely především hry pro tyto pokročilé platformy. Rozvoj byl také způsoben možností používat externí paměťové karty, čímž se zvýšila velikost úložného prostoru pro instalované aplikace. V této edici bylo rozlišení displeje pevně stanoveno na 176x208 px a jiné odchylky nebyly povoleny [42], [3].

- Symbian OS 7.0 byl poprvé uveden v roce 2003. Nově byl podporován rychlejší přenos dat EDGE. Podpora Javy byla změněna z pJava a JavaPhone na verzi Java ME Standard. Kódové označení OS 7.0 odpovídalo S60 2nd Edition a později S60 2nd Edition Feature Pack 1. Tyto verze byly mezi sebou zpětně kompatibilní a přinesly rozšíření o používání uživatelských profilů prostředí, pokročilý internetový prohlížeč, možnost přehrávat MP3 soubory. Nejdůležitější rozšíření až ve verzi 7.0s bylo rozšíření Java vlastností, nazvané MIDP 2.0.

  Také se na této platformě objevil první virus pro telefony se Symbian OS Cabir. Na jiné telefony se šířil automaticky pomocí rozhraní Bluetooth [42], [3].

- Následující verze Symbian OS 8.0 představená v roce 2004 umožňovala výběr mezi dvěmi jádry (EKA1 nebo EKA2). EKA1 je označení pro původní jádro, udržující zpětnou kompatibilitu s ovladači zařízení napsanými pro Symbian OS v7.0s a dřívější. EKA2 je pak nová verze jádra s vylepšenou podporou multitaskingu a multithreadingu, nazývaná „hard real time kernel“. Nicméně, verze jádra EKA2 nebyla využita až do příchodu Symbian OS 8.1b. Jádra se chovala navenek zcela identicky a uživatel rozdíl nepoznal. Jedním z hlavních vylepšení je podpora sítí třetí generace při zachování podpory sítí 2.5G. Nově přibyla podpora CDMA, DVB-H a OpenGL ES s vektorovou grafikou. K významnému pokroku došlo také v podpoře Javy. Symbian OS 8.0 nyní podporuje J2ME MIDP 2.0 a CLDC 1.1, používá CLDC HI 1.1 Java VM technologie firmy Sun Microsystems a odpovídá standardu JTWI (JSR185). Technologie PersonalJava a JavaPhone byly vystřídaný CDC/PersonalProfile [42], [3].

  Verze 8.0a známá pod názvem S60 2nd Edition Feature Pack 2 měla velice důležitou inovaci. Jednalo se o možnost spuštění systému z NAND Flash paměti, což vyústilo v uvolnění většího množství RAM. Také bylo vylepšeno grafické rozhraní, byla přidána podpora OpenGL ES 3D grafiky a průhlednost oken.

- Vývojová verze Symbian OS 9.0 znamenala konec jádra EKA1 a tudíž konec kompatibility mezi novými a staršími verzemi. Z důvodu použití nové architektury jsou binární kódy nepřenositelné a tudíž starší programy bylo nutné upravit pro funkčnost na nových systémech.

  O proti předchozím verzím je rozšířena bezpečnost systému díky podepisování instalovaného softwaru. Je také nutné potvrdit přístup softwaru třetích stran k systémovému API. Byl přidán nový vylepšený kompilátor, založený na Embedded Application Binary Interface (EABI)[4], a rozšířena práce s multimédii. Do této verze se řadí v současné době nejpoužívanější typy OS Symbian, jedná se o S60 3rd Edition až po S60 3rd Edition Feature Pack 2. Označení Feature Pack reprezentuje v podstatě to, co Servicepack pro OS Windows na platformě PC, či jiné plnohodnotné softwarové produkty, které opravují chyby a přidávají nové funkce. Rozdíly mezi jednotlivými Feature Packy jsou pro běžného uživatele hlavně v grafickém zpracování obrazovky a jejím nastavení.

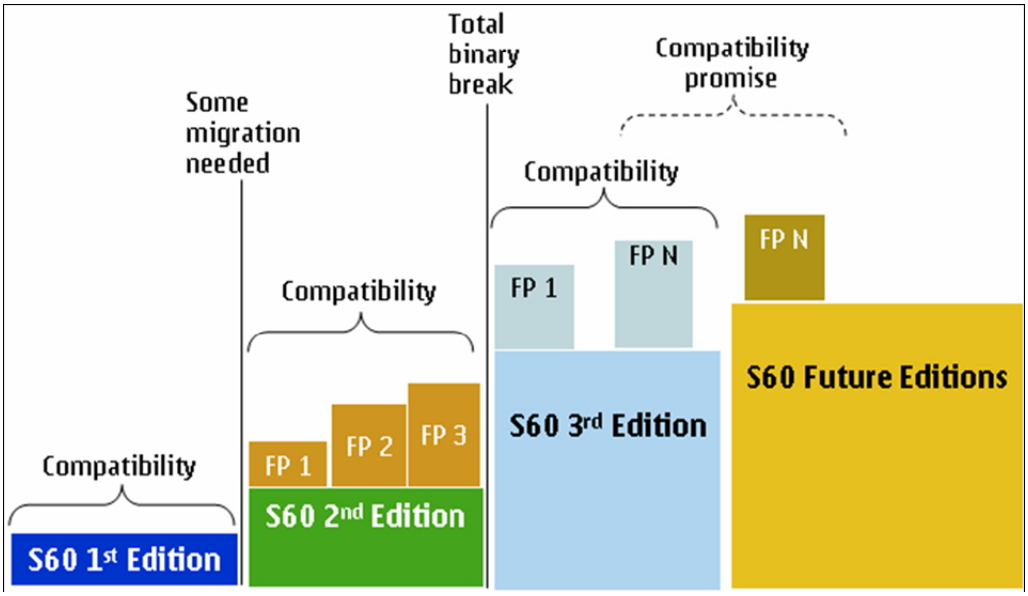
[4] popisuje nízkoúrovňové rozhraní mezi aplikacemi a operačním systémem, aplikacemi a jejich knihovnami nebo součástmi aplikací.

Z pohledu vývojáře je zlepšeno a rozšířeno využití programovacího jazyka C++ o C standard. Značné rozšíření jazyka Java o bezpečnostní a grafické funkce. Lepší využívání ROM paměti vede k urychlení práce systému i programů třetích stran [42], [3].

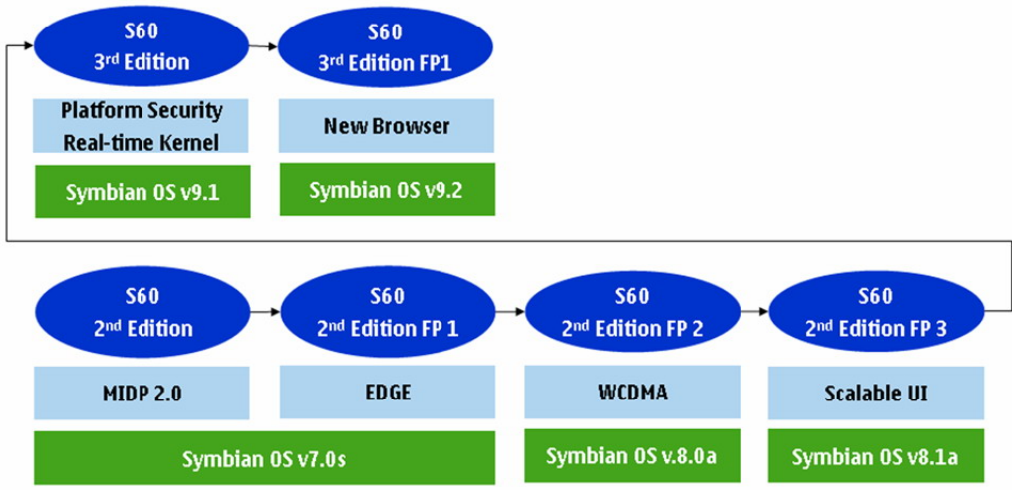
- Poslední oficiálně vydanou a používanou verzí je Symbian OS 9.5 pojmenovaný S60 5th Edition (což odpovídá novému označení Symbian[1]). Ta do značné míry staví na multimediálních a internetových zkušenostech z předchozích edic. V posledním release OS navíc implementuje doteková uživatelská rozhraní, senzorové technologie a rozšiřuje podporu různých rozlišení, širokoúhlého režimu i open source vývojářů (standard C/C++ API) [11].

- V současné době je připravován nový operační systém s názvem Symbian platform a označením Symbian[3][30]. Jde o nástupce Symbian OS. Symbian platform byl představen jako open source kód v březnu 2010, což je oproti minulým verzím zásadní změna, dovolující programátorům větší využití potencionálu tohoto OS. Symbian platform je rozdělen do více balíčků. Každý reprezentuje jednu technologickou doménu, obsluhující jistou funkcionalitu systému. Podle posledních informací by neměl být zpětně kompatibilní s dřívějšími systémy.

  Na obrázcích 2.2, 2.3 je graficky znázorněna historie a vývoj operačního systému Symbian. Obrázek 2.2 popisuje kompatibilitu mezi jednotlivými vývojovými verzemi. Vyplívá z něj i slíbená kompatibilita do budoucích vývojových verzí, která byla s příchodem Symbian[3] porušena. Druhý obrázek 2.3 ukazuje hlavní vývojové rozdíly mezi jednotlivými verzemi, počínaje Symbian S60 2nd Edition a konče Symbian S60 3rd edition FP1.



Obrázek 2.2: Historie Symbian OS [11].



Obrázek 2.3: Vývoj Symbian S60 [11].

2.3 Vývoj v prostředí Symbian S60

Vývoj pro prostředí Symbian S60 má svá jistá specifika. Jedná se o vývoj za využívání různých SDK a pluginů, které rozšiřují SDK o další funkcionalitu. Každý vývojový stupeň S60 má své speciální SDK, které využívá veškeré prostředky odpovídající dané vývojové verzi. Při vývoji aplikací využívající speciální funkce mobilního telefonu, jako je akcelerační čidlo nebo GPS modul, není zaručena úplná kompatibilita mezi různými SDK. Tímto se komplikuje vývoj specializovanějších programů pro více cílových platforem zároveň.

2.3.1 SDK

Zkratka SDK odpovídá anglickému výrazu Software development kit. Jedná se o softwarový vývojový balíček obsahující vývojářské nástroje, které umožňují vytvářet aplikace pro jistý softwarový framework, hardwarovou platformu, operační systém, atd.

Většinou se jedná o skupinu rozhraní pro programování aplikací (zkratka API z anglického Application Programming Interface), které obsahují procedury, funkce či třídy knihoven. API určuje, jakým způsobem se funkce knihovny volají ze zdrojového kódu programu. SDK je distribuováno společně s vývojovým prostředím, případně jako jeho zásuvný modul.

Protože jednotlivé verze Symbian OS nejsou mezi sebou 100% kompatibilní, je nutné objasnit, které SDK je možné použít. Jelikož podpora pro akcelerační senzor je podporována až od verze platformy S60 3rd Edition. Je využití starších verzí nemožné. Pro vývoj aplikace s využitím integrovaných senzorů jsou tedy k dispozici SDK S60 3rd Edition FP1, S60 3rd Edition FP2 a S60 5th Edition [26], [22].

2.3.2 Platform security

Platform security je označení pro bezpečností model Symbian OS, který je jeho součástí od verze 9. Jedná se o řadu oprávnění označených jako *capabilities*. Tyto oprávnění dovolují programům přistupovat k jinak zakázaným, případně chráněným, funkcím systému. Jistá povolení, která nezasahují do chodu systému, může povolit i uživatel mobilního telefonu, při instalaci aplikace. V jiném případě je nutné instalační soubor aplikace digitálně podepsat. Autoritou pro udělování digitálních podpisů je internetový portál *symbiansigned.com* [27]. Digitální podpisy jsou určeny jak pro komerční projekty, tak pro vývoj volně šířitelných programů. Základní rozdíl mezi jednotlivými licencemi je v počtu cílových přístrojů, na které se daná aplikace bude instalovat. V případě vývojářské licence, která je zdarma, je počet přístrojů omezen pouze na jeden. V případě rozšíření aplikace na více přístrojů, je nutné pro každý tento přístroj zažádat o novou licenci [31].

Kapitola 3

Události v mobilních telefonech

Následující text se zaobírá problematikou předávání zpráv v operačních systémech Symbian běžících na mobilních telefonech firmy Nokia. Jde především o přejímání zpráv o událostech při přijetí, odesílání, respektive modifikaci textových zpráv, hovorů a obsluhování zabudovaných zařízení, mezi něž se například počítá GPS modul nebo akcelerační čidlo. Nejsou zde řešeny jednotlivé rozdíly mezi různými programovacími jazyky, ale je snaha o obecné popsání problematiky komunikace v systémech Symbian OS S60.

3.1 Hardware mobilních telefonů

Uživatelské rozhraní Symbianu S60 (dále jen S60) bylo původně navrženo pro používání jednou rukou. Z toho důvodu je na mobilních telefonech S60 ve většině případech nabízena standardní klávesnice ITU-T[1], tlačítka pro přijetí a ukončení hovoru a vícesměrná pohybová klávesa. Omezený počet kláves se projevuje v uživatelském rozhraní a aplikace se musí vyvíjet s tímto ohledem.

Hardwarová architektura S60 mobilních telefonů je v současné době založena spíše na fixed-point digital signal procesorech (DSP) než na mikroprocesorech počítající v plovoucí čárce, které jsou většinou umístěny ve stolních počítačích. Je to z důvodu, že mobilní telefony v základě pracují s video a audio daty a jako přenosná zařízení musí také šetřit spotřebovanou energií. Typická rychlost procesorů je mezi 100 až 600 MHz, což splňuje požadavek na malou spotřebu a předchází velkému vyzařování tepla. Jako nevýhoda se zřejmě ukazuje malý výpočetní výkon pro použití sofistikovaných algoritmů a grafických aplikací. I když operační systém Symbian dovoluje použití floating-point operací, nedoporučuje se jejich používání z důvodu malé efektivnosti, rychlosti a velké spotřeby energie ve srovnání s výpočty s pevnou desetinnou čárkou.

Paměť je v mobilních telefonech většinou omezená na několik megabytů pro aplikace, u starších telefonů přibližně megabyte pro paměť typu halda [2] (paměť dostupná pro běžící aplikace). Omezení paměti typu hromada na několik MB je u nových telefonů odstraněno a využívá se celý dostupný paměťový prostor. Také pokud se předpokládá přenos aplikací bezdrátově, je nutné si ověřit maximální možnou velikost přenášeného souboru. Mobilní telefony poslední generace, již mají paměť o několik desítek megabytů větší, ale i přesto však nedosahuje kapacit stolních počítačových systémů [3], [4], [5].

[1] Sektor Mezinárodní telekomunikační unie - Normalizace v telekomunikacích

[2] v originále Heap memory

3.2 Uživatelské vstupy

Nejjednodušší cestou, jak uživatel může zadat požadavek mobilnímu telefonu je přes jeho klávesnici. Klávesnice na mobilním telefonu je většinou rozdělena na 3 části:

- klávesnice (obsahující klávesy 0-9, * a #)
- „soft keys“ (tlačítka pro přijetí a ukončení hovoru)
- vícesměrná pohybová klávesa

Klávesnice může být použita jak pro numerické, tak pro textové zadávání. „Soft keys“ se používají pro otevírání menu, výběr položek v menu a ukončování aplikací. Jejich funkce je většinou potvrzení a odmítnutí akce, při čemž se radí zachovávat stejnou funkci jak při přijímání a ukončování hovoru. To znamená levá klávesa značí pozitivní akci, pravá naopak negativní akci. Přímo v knihovnách existují připravená nastavení těchto kláves. Vícesměrná pohybová klávesa umožňuje pohyb kurzorem a její stlačení funguje jako tlačítko výběru, nebo OK klávesa.

Protože mobilní telefon se používá pro hlasové služby, jsou snahy o zvýšení úlohy hlasu v ovládání. Tento vstup však není zdaleka tak využit jak by si zasloužil a to ani na stolních počítačích. Většinou se nedá využít z důvodu nedostatečného výkonu procesorů a chybějících podpůrných aplikací.

Další okruh vstupních zařízení otevírají bezdrátové technologie jako je Bluetooth, přes něj lze připojit obrovské množství různých příslušenství, například klasickou počítačovou klávesnici [1].

3.2.1 Události kláves

Obsluha kláves je prováděna pomocí událostí. Jsou rozpoznávány tři druhy událostí, reakce na stlačení klávesy (key down), na puštění klávesy (key up) a standardní stisk klávesy. Ve struktuře popisující událost je obsažena informace o tom, která klávesa byla zmáčknuta, stav modifikované klávesy a pozice kurzoru, počet opakování a scan kód [1].

3.2.2 Události dotykového displeje

Ovládání pomocí dotykového displeje je založena na zasílání zpráv o událostech. Události jsou vyhodnocovány obdobně jako události kláves a obsahují reakci na stlačení obrazovky, na puštění obrazovky a standardní stisk obrazovky. Speciální stisky obrazovky jsou registrovány pomocí Touch UI observerů. Jedná se o:

- Navigační observer - je použit v navigační části obrazovky. Rozeznává jestli byla zmáčknuta grafická reprezentace levé nebo pravé šipky.
- Detektor dlouhého zmáčknutí - jedná se o detektor dlouhého zmáčknutí dotykové obrazovky, kdy zmáčknutí trvá déle než 0.15s. Je oznamováno vykreslením animace znázorňující dlouhý stisk.
- Observer title panelu - reaguje na události v title panelu [1].

3.2.3 Komunikace

Aktivní objekty

V S60 je komunikace mezi zařízeními řešena asynchronně. Aktivní objekty jsou řešením, které se stará o paralelní asynchronní operace používajících pouze jedno vlákno. Každá aplikace je prováděna pouze ve svém vlastním vláknu. Aktivní objekty jsou odpovědné za požadavky. K tomu využívají aktivní plánovač, který se stará o zpracování událostí podle jejich prioritních uspořádání. Každé vlákno, které může obsahovat více aktivních objektů, obsahuje jeden aktivní plánovač. Přístup využívající aktivní objekty je doporučený a efektivnější než využívání vláken, díky menší režii při přepínání jednotlivých asynchronních operací [1].

Sériová komunikace

Sériová komunikace probíhá na způsobu architektury klient-server. Pro vytvoření sériové komunikace je nutné nejprve vytvořit sezení se serverem. Server poskytne klientovi informace o protokolech pro danou službu a také port, kde je tato služba přístupná. Dovoluje nastavení rychlosti přenosu, množství dat a paritní bity atd [1].

Textová komunikace

Architektura textové komunikace poskytuje framework pro posílání a přijímání zpráv pomocí protokolu jako je SMS nebo MMS. Funkčnost pro každý komunikační protokol je poskytnuta skrze několik rozdílných typů Machine-To-Machine (MTM) modulů, přičemž některé z nich poskytují API funkce pro daný protokol. SMS klient MTM modulu poskytuje operace pro posílání a přijímání textových zpráv. Také se používá pro uložení nastavení SMS klienta MTM modulu a podává informace o aktuálním stavu odesílání nebo přijímání zprávy [1].

Multimediální komunikace Multimediální komunikace je v mobilních telefonech řešena pomocí multimediálních zpráv (zkratka MMS z anglického Multimedia Messaging Service). Pomocí MMS je možné posílat kromě textu i obrázky, audio a videoklipy. MMS zprávy jsou přenášeny jako datový tok prostřednictvím datové technologie GPRS a pokročilejších technologií. Pro přenos obrázku se u MMS používají nejčastěji formáty GIF, PNG nebo JPG. Pro přenos zvuku AMR nebo WAV a pro video MPEG-4.

Pro MMS komunikaci je v Symbian OS určeno MMS Client MTM API. Toto API umožňuje vytvářet a odesílat multimediální zprávy. Také dovoluje zprávy přijímat a upravovat záznamy multimediálních zpráv v Message Serveru. Multimediální obsah je při odesílání připojen ke zprávě. Potřebné informace o reprezentaci zprávy jsou definovány značkovacím jazykem SMIL[3]. Manipulace se zprávami probíhá pomocí MTM modulů, které jsou popsány v části Textová komunikace [29].

Komunikace pomocí socketů

Sockety poskytují jednoduché mechanizmy pro spojování (Symbianovských) aplikací s externími zařízeními. Sockety podporují architekturu komunikace klient-server, za pomoci Symbianovské API pro komunikaci se serverem. V API je implementováno také přiřazování jmen, speciálně pro IP a Bluetooth. Dále je obsaženo velké množství tříd starajících se

[3] http://www.w3.org/AudioVideo/

o ukládání jmen a protokolových informací potřebných pro komunikaci a používání socketového spojení. Komunikace probíhá pomocí čtení a zápisu informací na ustanoveném spojení [1].

Infračervená komunikace

Symbian OS podporuje komunikaci přes infračervené světlo používající IrDA standard. Pokud je zařízení v dosahu přenosového čidla, dojde k servisní komunikaci mezi jednotlivými zařízeními a použití protokolu, který povolí přenos dat. IrDA umožňuje komunikovat jak pomocí socketů, tak pomocí sériové komunikace.

Pro oba přístupy je implementována speciální komunikace pro přijímání obrázků z fotoaparátů obsahujících IrDA port a monitorování průběhu přenosu. Obrázky jsou přenášeny pomocí Unified Picture formátu, jenž podporuje uložení do formátu JPEG [1].

Bluetooth komunikace

Bluetooth je krátkorozsahová bezdrátová technologie, která zajišťuje vysokou úroveň bezpečnosti a spolehlivosti za cenu malé spotřeby energie. Pracuje na frekvencích mezi 2.4 a 2.4185 GHz, používající rozdělení signálu pomocí časového duplexu[4]. Rozsah signálu je v rozmezí 10 až 100 metrů, v závislosti na mobilním telefonu. S60 poskytuje Bluetooth komunikaci přes sockety a Bluetooth API, které přistupuje k Bluetooth zásobníku Symbian OS [1].

Z hlediska implementace jsou zajímavé následující softwarové vrstvy:

- Service Discovery Protocol (SDP) dovoluje klientovi zjistit služby nabízené zařízením vystupujícím jako server. Slouží také k detekci ukončení Bluetooth služby.
- Radio Frequency Communications (RFCOMM) emuluje sériový port pro L2CAP protokol.
- Logical Link Control and Adaptation Protocol (L2CAP) poskytuje jednotnou správu přenosu ke skupině jiných Bluetooth přístrojů a rozdělování a znovu spojovaní paketů.

IP komunikace Pod pojem IP komunikace se v Symbian OS řadí TCP/IP přenos a IP aplikační protokoly.

O fungování TCP/IP přenosů se stará IP Connection Management (ipconnmgmt), který má funkci řízení a monitorování TCP/IP přenosu. Poskytuje infrastrukturu, nastavení uživatelské rozhraní, algoritmy pro multi-homed[5] TCP/IP a službu pro PDP context (Packet Data Protocol context)[6]. Přístupové body k internetu a cílové síti jsou spravovány pomocí API poskytované balíčkem ipconnmgmt [33].

IP aplikační protokoly obsahují protokoly jako SIP[7] a RTP[8]. Ty jsou nutné pro správu multimediálních IP relací, například VoIP[9] relací.

SIP Framework poskytuje API pro enkódování, dekódování a nastavení vlastností SDP[10] transakcí. Taktéž obsahuje API pro správu SIP přenosů a komunikací, podporuje bezpečností mechanizmy a komprimaci dat. Obsahuje SigComp (Signaling Compression) poskytující služby pro komprimování a dekomprimování textových protokolů jako jsou SIP, SDP

[4] v originále TDD (Time Division Duplex)
[5] stav, kdy hostitelský uzel je připojen do více sítí nebo má více síťových adres.
[6] logické spojení mezi GPRS účastníkem a externí IP sítí.
[7] protokol pro inicializaci relací
[8] Real-time Protocol
[9] Voice over Internet Protocol
[10] přenosový protokol

a RTSP[11]. SIP Framework dokáže pracovat s datovými sítěmi provozovanými mobilními operátory i se sítěmi typu Wi-Fi. Obsahuje grafické uživatelské rozhraní pro správu SIP profilů a dokáže monitorovat průběh komunikace [32].

Instant Messaging komunikace Instant messaging (zkratka IM)je internetová služba, umožňující svým uživatelům sledovat, kteří jejich přátelé jsou právě připojeni a podle potřeby jim posílat zprávy. Jedná se o posílání zpráv v reálném čase.

V Symbian OS je IM komunikace spravována IM API. Obsahuje správce připojení a správce IM komunikace. Správce připojení se stará o připojení a o získávání informací o spojení. Správce IM komunikace odesílá a přijímá IM zprávy a zjišťuje případné chyby v IM komunikaci (například neexistující uživatel). IM API je implementováno jako aktivní objekt a tudíž neblokuje ostatní operace. Ke komunikaci se využívá IMPS[12] protokol [28].

3.3 Čidla a moduly

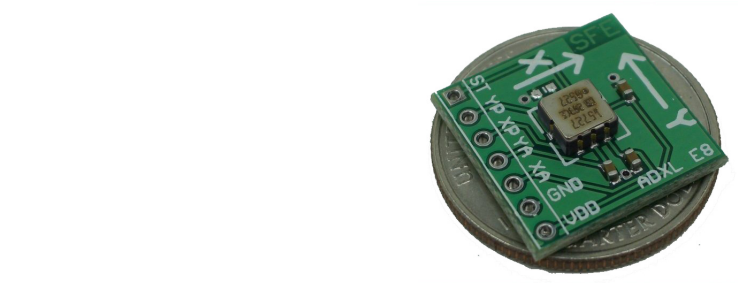
Mobilní telefony sloužící nejen k telefonním účelům obsahují množství rozšiřujících funkcí. Mezi největší lákadla patří přehrávání MP3 souborů, video souborů, používání bezdrátového internetu přes Wi-Fi připojení, GPS navigace, ovládání mobilního telefonu pomocí pohybu a další rozšíření.

Většina výše zmíněných rozšíření má již naimplementovanou širokou podporu v mobilních aplikacích. GPS navigaci podporují veškeré velké firmy zabývající se touto problematikou (například Garmin, TomTom) a také firmy vyrábějící mobilní telefony mají vlastní řešení navigace. Akcelerační čidlo je využíváno pro ovládání her, přehrávačů i osobních počítačů. Avšak v málo případech jsou tyto čidla a moduly spojeny v jednom programu.

3.4 Akcelerační a světelné čidlo

Akcelerační čidlo, neboli akcelerometr, je senzor, který využívá setrvačnosti hmoty pro měření rozdílu mezi kinematickým zrychlením (vzhledem k určitému inerciálnímu prostoru) a gravitačním zrychlení. Trendem v oblasti vývoje je příklon k MEMS (mikroelektromechanickým) akcelerometrům. Klasické mechanické senzory jsou tak v dnešní době nahrazovány součástkami vyrobenými touto MEMS technologií, které mají mnohem menší rozměry, nižší energetickou spotřebu a podstatně nižší cenu. Nevýhodou těchto součástek je zatím stále nedostatečná přesnost pro mnohé aplikace [41]. Na obrázku 3.1 je zobrazen akcelerometr LIS302DL používaný v mobilních telefonech Nokia.

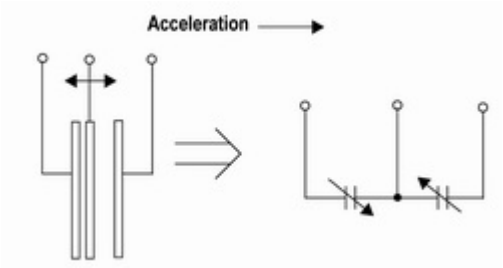
[11] Real-time Transport Protocol
[12] Instant Messaging and Presence Services



Obrázek 3.1: Akcelerometr LIS302DL [52].

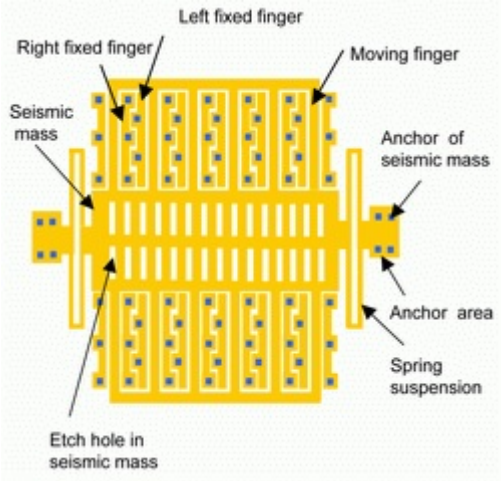
Princip MEMS akcelerometru je založen na proměnné kapacitě tříelektrodového vzduchového kondenzátoru. Využívá se zde nelineární závislosti kapacity C na vzdálenosti elektrod kondenzátoru d dle vzorce $C = \varepsilon_0 \varepsilon_r \cdot \frac{S}{d}$, kde $\varepsilon_0 \varepsilon_r$ jsou konstanty, S je plocha elektrod. Pokud tedy jednu elektrodu uděláme pohyblivou a její pohyb bude závislý na působícím zrychlení, získáme kapacitní akcelerometr [36].

Výsledná funkční struktura, znázorněná na obrázku 3.2, je závislá na zajištění lineárního a dostatečně citlivého převodu zrychlení na mechanický posuvný pohyb. Ten totiž určuje samotný měřící rozsah senzoru, tj. maximální a minimální měřitelné zrychlení.



Obrázek 3.2: Základní princip MEMS akcelerometru [36].

Vychází ze základního vztahu pro působení síly při zrychlení $F = m \cdot a$, kde F je síla vzniklá působením zrychlení a na hmotu m. Síla se pak přes pružiny převádí na posuv pohyblivých elektrod vzduchového kondenzátoru. Jejich pozice vůči levým pevným elektrodám a pravým pevným elektrodám určuje elektronicky měřenou hodnotu kapacity takto vzniklého kondenzátoru. Grafické znázornění MEMS struktury akcelerometru je na obrázku 3.3.



Obrázek 3.3: Schématicky znázorněná mechanická MEMS struktura akcelerometru [36].

Výše uvedená a popsaná struktura však umožňuje měření zrychlení jen v jednom směru kolmém na pohyblivé elektrody = 1D akcelerometry. Pro vznik 2D akcelerometru je nutné na čip přidat další stejnou strukturu pouze proti té předchozí pootočenou o 90°. Složitější je již vytvořit jednočipový 3D akcelerometr, protože se musí přidat výškově pohyblivá struktura v ose Z [36].

Akcelerační čidlo je většinou od výrobce používáno jen pro správnou orientaci zobrazení obsahu displeje a pro zjištění orientace vyfotografovaných obrázků, pro jejich správné natočení a následné uložení. Aplikace třetích stran, využívající akcelerometr, jsou většinou hry, které natáčení používají pro vlastní ovládání. Smysluplných programů využívajících čidlo je velice málo a jsou to většinou programy vyrobené pro nekomerční účely. Nejznámější program, kde je použito akcelerační čidlo, je FlipSilent. Tento program využívá pohyb na ovládání příchozích hovorů.

Světelné čidlo je v mobilních telefonech Nokia se Symbianem S60 využíváno pouze k nastavení intenzity podsvícení displeje. Je to z důvodů, že jeho programové využití nebylo do verze S60 3rd edition FP2 možné.

Obrázek 3.4: Světelné čidlo [54].

Světelné čidlo funguje na principu průchodu proudu fotodiodou. Fotodioda je plošná polovodičová dioda konstrukčně upravena tak, aby do oblasti PN přechodu[13] pronikalo světlo. Není-li přechod osvětlen, má voltampérová charakteristika fotodiody stejný průběh, jako charakteristika běžné diody.

Fotodioda je založena na vnitřním fotoelektrickém jevu. Světlo (foton), který dopadá na přechod PN, narazí do elektronu ve valenční vrstvě atomu a předá mu svoji energii. Elektron energii fotonu absorbuje, čímž získá dostatek energie k opuštění valenčního pásu a přeskočí do pásu vodivostního. Takto vzniklé volné elektrony jsou volné nosiče náboje, které snižují elektrický odpor polovodiče, resp. zvyšují elektrickou vodivost polovodiče. V závislosti na zapojení fotodiody dojde ke zvýšení, nebo snížení vodivosti na přechodu PN [45].

3.4.1 Přístup k akcelerometru v OS Symbian S60

Pro přístup k senzoru se v Symbian S60 používá více metod. Každá metoda pracuje na jiné vývojové verzi Symbian S60.

Nejdokonalejším přístupem je využití frameworku S60 Sensor Framework, který byl primárně vytvořen pro platformu S60 5th Edition. Pro předchozí platformy byl upraven a jeho funkčnost se tedy rozšiřuje i na verze S60 3rd Edition Feature Pack 2 a také pro mobilní telefon Nokia E66 (S60 3rd FP1). Tato verze není kompatibilní s předchozími API frameworky a je tedy nutné nejprve určit cílovou platformu, na které má výsledný program pracovat. Aplikace Sensor Frameworkem vytvořená může přistupovat k více čidlům zároveň, tím je zajištěna alespoň částečná kompatibilita mezi různými zařízeními obsahujícími S60 Symbian OS. K jednomu čidlu může také přistupovat více aplikací zároveň.

Druhý přístup je využitím Sensor API. Toto API se využívá ve verzích S60 3rd Edition a S60 3rd Edition Feature Pack 1. Dovoluje přistoupit jen k omezenému množství čidel [19].

S60 Sensor Framework APIs

Existují tři hlavní API, které se používají v S60 Sensor Framework.

- Sensor Plug-in API - jde o vnitřní S60 API, které není obsaženo ve volně šířitelném SDK. Je určené pro S60 licencované produkty a obsahuje více pluginů pro datové komunikace s čidlem.
- Channel Finder API - se používá pro zjištění komunikačních kanálů pro jednotlivá zařízení. Může zpřístupnit všechna, nebo jen určitá data ze senzoru.
- Sensor Channel API - slouží pro komunikaci se senzorem. Má následující funkce:
  - Metody pro otevírání a uzavírání kanálů.
  - Metody pro započetí a u končení odposlouchávání dat ze senzoru.
  - Metody pro nastavení a zjišťování nastavení na datovém kanálu senzoru.

Framework má přístup přímo k datům z čidla, tudíž je možné využít všechny informace jím dodávané pro jakékoliv zpracování bez ztráty informací a omezení daných operačním systémem [19].

Nokia Sensor APIs

Nokia Sensor API je přístupné přes Sensor plug-in pro S60 SDK a dovoluje získávat data z vestavěných senzorů. Obsahuje následující metody:

[13] oblast na rozhraní polovodiče typu P a polovodiče typu N

- Metody pro zjištění počtu aktivních senzorů.
- Metody pro započetí a u končení odposlouchávání dat ze senzoru.
- Metody pro nastavení a zjišťování nastavení na datovém kanálu senzoru.

Sensor API má také přístup přímo k datům z čidla, takže informace z čidel jsou doručeny bez jakýchkoliv úprav [19].

3.4.2 Přístup ke světelnému senzoru v OS Symbian S60

Jak bylo zmíněno výše, přistup ke světelnému čidlu je omezený. Pro mobilní telefony obsahující OS S60 3rd Edition FP2 a novější je možné využít S60 Sensor Framework API 3.4.1 a přistupovat k němu jako k jiným senzorům. U starších verzí obsahujících i OS S60 3rd Edition FP1 není možné k tomuto senzoru přistoupit a využívat jeho informace.

V případě potřeby použit informace o okolním světle v aplikaci, je jediná možnost toto čidlo vynechat a k analýze obrazu využít jinou metodu. Jako jediná možná metoda se jeví použití fotoaparátu, nejlépe toho, který je umístěn na čelní straně mobilního telefonu. Vytvoření fotografie a její následné analýzy na intenzitu světla. Více informací o fotografickém čipu je v kapitole 3.5.

3.5 Fotografický senzor

Možnost fotografování mobilními telefony je v poslední době brána jako samozřejmá. Náklady na výrobu fotografických senzorů je tak nízká, že i mobilní telefony nejnižších tříd obsahují jednoduchý senzor. Výstupy těchto senzorů jsou sice na velice nízké úrovni, ale pro zachycení momentek úplně dostačují. U vyšších řad mobilních telefonů jsou senzory kvalitnější a v jistých případech dosahují kvalit kompaktních fotoaparátů.

Pro získávání digitálních fotografií se ve většině případu používají dva druhy senzorů. CCD nebo CMOS senzor. Oba senzory však zachytávají pouze intenzitu světla a barevnost se dociluje pomocí Bayerovy masky. Rozdíl mezi CCD a CMOS senzorem je ve způsobu sběru signálu z buněk a v logice ovládání senzoru.

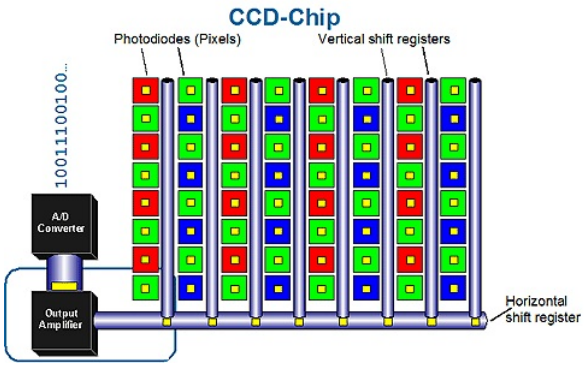
Buňka je základní stavební prvek senzoru. Rovnoměrně pokrývají jeho plochu. Buňky však nejsou zcela na povrchu senzoru, nýbrž v malých jamkách. To je jednak vyvoláno technologickými potřebami, ale také to omezuje vzájemné ovlivňování buněk mezi sebou a tím zvyšuje obrazovou kvalitu. Buňky také nemohou pokrýt celou plochu oblasti, která je jim teoreticky vyhrazena. Plochu sdílí s vodiči a další elektronikou, například tranzistory. Pro vyřešení obou těchto problémů je před senzorem pole mikroobjektivů. Ty usměrňují světlo do jamek a také soustřeďují světlo z větší plochy na menší plochu aktivní části buňky.

Senzor je analogové zařízení (výstupem senzoru je analogové napětí). Je nutné za něj přidat A/D převodník[14], jenž převede napětí na hodnoty, které je možné dále zpracovat.

- CCD senzor (Charged Coupled Device) datuje svojí historii již od roku 1969 a využívá svoji schopnost transportovat signál z buněk skrze jiné buňky, aniž by tím utrpěla kvalita signálu. Takto se signál postupně posouvá až na okraj, kde je posuvný registr, který potom signál jeden po druhém předá do zesilovače a A/D převodníku. Posun signálu skrze buňky probíhá díky nábojové vazbě buněk a tato schopnost dala senzoru i své jméno Charged Coupled - svázány nábojem.

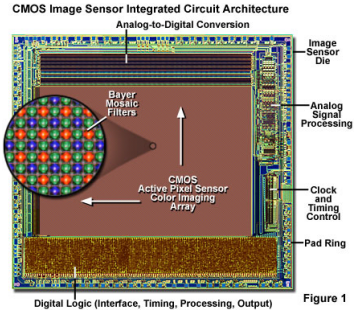
[14] Analogově digitální převodník

CCD senzor používá unikátní metodu výroby a to jinou než ostatní integrované obvody (například paměti nebo procesory). Proto je poněkud problém dosáhnout jeho velkého rozlišení. Ze stejného důvodu je rovněž problematické integrovat do CCD senzoru jinou elektroniku a tak většina řídících obvodů včetně zesilovačů a A/D převodníků je mimo senzor. Digitálního obrazu se tak dosáhne až pomocí dalších integrovaných obvodů na desce s plošnými spoji a CCD senzor také vyžaduje větší rozsah různého napájení, což opět komplikuje použití v reálném zařízení. Soubor integrovaných obvodů jako celek má potom i výrazně vyšší spotřebu ve srovnání s technologií CMOS [25]. Nákres CCD senzoru lze vidět na obrázku 3.5.



Obrázek 3.5: CCD senzor [53].

- CMOS (Complimentary Metal Oxide Semiconductor) senzor využívá v principu stejnou technologii výroby jako ostatní integrované obvody - procesory, paměti atd. Z tohoto důvodu je levnější, umožňuje vyšší stupeň integrace a není problém přímo v senzoru integrovat celou řadu dalších obvodů. Na rozdíl od CCD má v CMOS senzoru každá jednotlivá buňka svůj vlastní zesilovač a může být díky tomu přímo adresována a čtena pomocí jejích X,Y souřadnic. Zásadní rozdíl tedy není ve vlastní konstrukci citlivé části buňky (fotodiodě), ale v tom, jak je buňka čtena. U CCD jsou buňky čteny postupně díky přenosu náboje skrze buňky, u CMOS je každá buňka podobně jako u běžných pamětí či LCD obrazovek samostatně adresována pomocí jejich souřadnic [25]. Princip CMOS senzoru je znázorněna na obrázku 3.6.



Obrázek 3.6: CMOS senzor [55].

V mobilních telefonech, které jsou náchylné na velký odběr energie a cenu součástí, je využíván CMOS senzor. Díky své malé spotřebě a ceně převážil nevýhodu většího šumu.

3.5.1 Přístup k fotografickému senzoru v OS Symbian S60

Operační systém Symbian S60 má možnost programově přistupovat k fotografickému senzoru za využití **Camera Application Engine API**. Toto API ovšem není standardně obsaženo v SDK pro Symbian S60 a je nutné ho doinstalovat z **SDK API Plug-in** který je volně ke stažení[15].

Interface **Camera Application Engine API** dovoluje zachytávát jak fotografie, tak i video. U fotografií podporuje různé režimy záznamu, nejvíce používané zachytávání stálých fotografií a také pořizování sekvenčních snímků. U nahrávání videa jsou jeho vlastnosti závislé na použitém SDK a programovacím jazyku. Přístupu k fotografickému senzoru je asynchronní a dovoluje tedy mít na pozadí spuštěné jiné aplikace, nebo procesy. Obsahuje metody na

- Inicializaci kamery.
- Inicializaci kamery pro nahrávání videa.
- Rezervování kamery pro použití v aplikaci. Pokud jiná aplikace používá fotografický čip, nelze kameru použít.
- Rezervování kamery pro nahrávání videa.
- Zapnutí kamery.
- Pořízení snímku.
- Spuštění nahrávání videa.
- Zastavení nahrávání videa.
- Uvolnění kamery a její vypnutí.
- Informování o jednotlivých stavech kamery [18].

3.6 Zvuk a jeho správa

Zvuk k mobilním telefonům patří už od jejich počátků. Využívá se ke komunikaci s jinými lidmi, při upozorňování na příchozí telefonáty a zprávy. Informuje o stavu mobilního přístroje a v neposlední řadě slouží k zábavě. Ve spoustě případů se využívá také jako přenosný přehrávač hudebních souborů a videoklipů. Při použití se speciálními programy se díky mikrofonu stává hlasovým záznamníkem nebo také slouží pro analýzu okolního hluku.

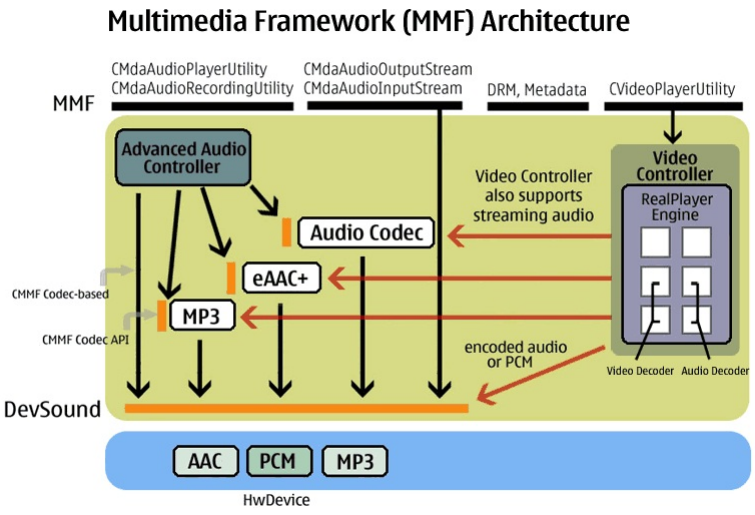
3.6.1 Přístup ke správě zvuku v OS Symbian S60

Přístup ke zvuku, ať už k nahrávání nebo přehrávání, je umožněn díky frameworku **Multimedia Framework** (zkráceně *MMF*). Framework se skládá z většího počtu API, která fungují jako plug-in moduly. Dají se rozdělit do dvou úrovní:

[15] http://wiki.forum.nokia.com/index.php/SDK_API_Plug-in

- **Nízkoúrovňové API**[16] používají MDF DevVideoRecord (pro přístup k video enkodérům) a DevSound (pro přístup k audio enkodérům a dekodérům). Většina těchto kodeků je urychlovány hardwarem (tzv. HwDevice). V nízkoúrovňovém API je bohužel jen omezený počet kodeků a v praxi se častěji používá vysokoúrovňové API [14], [15], [17].
- **Vysokoúrovňové API**[17] také známy jako MMF client APIs obsahují následující API:
  - CMdaAudioOutputStream a CMdaAudioInputStream pro přímý přístup k DevSound dekodérům.
  - CMdaAudioPlayerUtility používaný pro přístup ke všem audio dekodérům a souborovým formátům pro přehrávání. Jde o interface k MMF kontrolerům dostupných na daném zařízení. MMF kontroler umožňuje použít DevSound dekodér nebo CMMFCodec dekodér.
  - CMdaAudioRecordingUtility používaný pro přístup k audio enkodérům a souborovým formátům pro nahrávání. API je velmi podobné CMdaAudioPlayerUtility, jde také o interface k MMF kontrolerům a dovoluje použít DevSound enkodér nebo CMMFCodec enkodér.
  - CVideoPlayerUtility používané pro přístup k RealPlayer Engine obsluhující audio a video dekodéry.
  - CVideoRecorderUtility využívá MMF kontrolery pro nahrávání audia a videa z kamery zařízení a jeho následné uložení do souboru [14], [15], [17].

Jednotlivé uspořádání API je znázorněno na obrázku 3.7



Obrázek 3.7: Multimedia Framework architektura [14].

V každém z API je obsaženo několik základních metod sloužících k:

- Nahrávání/přehrávání zvuku a videa z/do různých datových úložišť (soubor, paměť)

[16] v originále „the low-level APIs“
[17] v originále „the high-level APIs“

- Volba kodeku zvuku a videa a jejich vlastností
- Ukončení nahrávání s případným uložením do souboru [14], [15], [17].

3.7 GPS modul

O GPS navigaci se s jejím rozmachem mluví čím dál častěji. Následující sekce popisuje informace o jejím použití.

3.7.1 Definice GPS

Global Positioning System, zkráceně GPS, je vojenský družicový systém pro určení polohy provozovaný Ministerstvem obrany Spojených států amerických, s jehož pomocí je možno určit polohu a čas kdekoliv na Zemi, nebo nad Zemí, s přesností několika metrů. Přesnost GPS lze s použitím dalších metod ještě zvýšit až na jednotky centimetrů. Část služeb tohoto systému s omezenou přesností je volně k dispozici i civilním uživatelům. V současné době se systém využívá v mnoha oborech lidské činnosti.

Uživatelé pomocí GPS přijímače přijímají signály z jednotlivých družic, které jsou v danou chvíli nad obzorem. Na základě přijatých dat (časových značek z jednotlivých družic a znalosti jejich polohy) a předem definovaných parametrů přijímač vypočítá polohu antény, nadmořskou výšku a zobrazí přesné datum a čas. Komunikace probíhá pouze od družic k uživateli, GPS přijímač je tedy pasivní [34].

Rozdělení přijímačů podle přijímaných pásem:

- jednofrekvenční - využívá pouze jedno frekvenční pásmo (L1), náchylné na zkreslení signálu procházející ionosférou
- dvoufrekvenční - využívá dvě frekvenční pásma (L1, L2) pro omezení zkreslení ionosférou
- vícefrekvenční - připravuje se pro pásmo L5, využívá více než jednu frekvenci

Rozdělení přijímačů podle kanálů:

- jednokanálové - dekódují data pouze z jedné družice.
- vícekanálové - pracuji současně se všemi družicemi nad horizontem.

Rozdělení přijímačů podle principu výpočtů:

- kódová - založena na zpracování kódového měření. Stanoví vzdálenosti jako součin doby a rychlosti šíření signálu mezi družicí a anténou.
- fázová a kódová - využívá kódového i fázového měření. Fázové měření je přesnější než kódové. Vzdálenosti mezi družicí a GPS aparaturou jsou určovány z měření nosné vlny GPS signálu. Při fázovém měření nesmí dojít k přerušení signálu. Jakékoliv i krátkodobé přerušení signálu znamená znemožnění určení správného celočíselného násobku vlnové délky a nutnost opakování měření [34].

Běžně dostupné přijímače k amatérskému (tj. negeodetickému a nevojenskému) využití se vyrábí jako jednofrekvenční, vícekanálové a kódové.

Jednoduchý přijímač signálu GPS pro se skládá z:

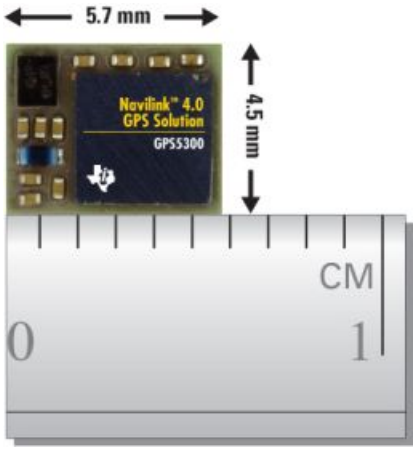
- antény
- předzesilovače
- procesoru
- časové základny (často křemíkový krystal o přesnosti ¡10-6s)
- komunikačního rozhraní

Uživatelé využívající systém GPS můžeme rozdělit do dvou skupin:

- **autorizovaní uživatelé** (vojenský sektor USA a vybrané spojenecké armády) využívající službu Precise Positioning Service (PPS) a mající k dispozici dekódovací klíče k P(Y) kódu na frekvencích L1 a L2. Tito uživatelé mají zaručenou vyšší přesnost systému [34].
- **ostatní uživatelé** (především civilní sektor) mohou využívat Standard Positioning Service (SPS) a mají k dispozici C/A kód na frekvencích L1. Přijímače vyrobené v USA nesmějí být exportovány, pokud nemají nastavená omezení výšky do 18 km a rychlosti do 515 m/s. Tyto limity vycházejí z prevence možného zneužití jako systému orientace v prostoru ve zbraních obdobných balistickým raketám nebo střelám s plochou dráhou letu [34].

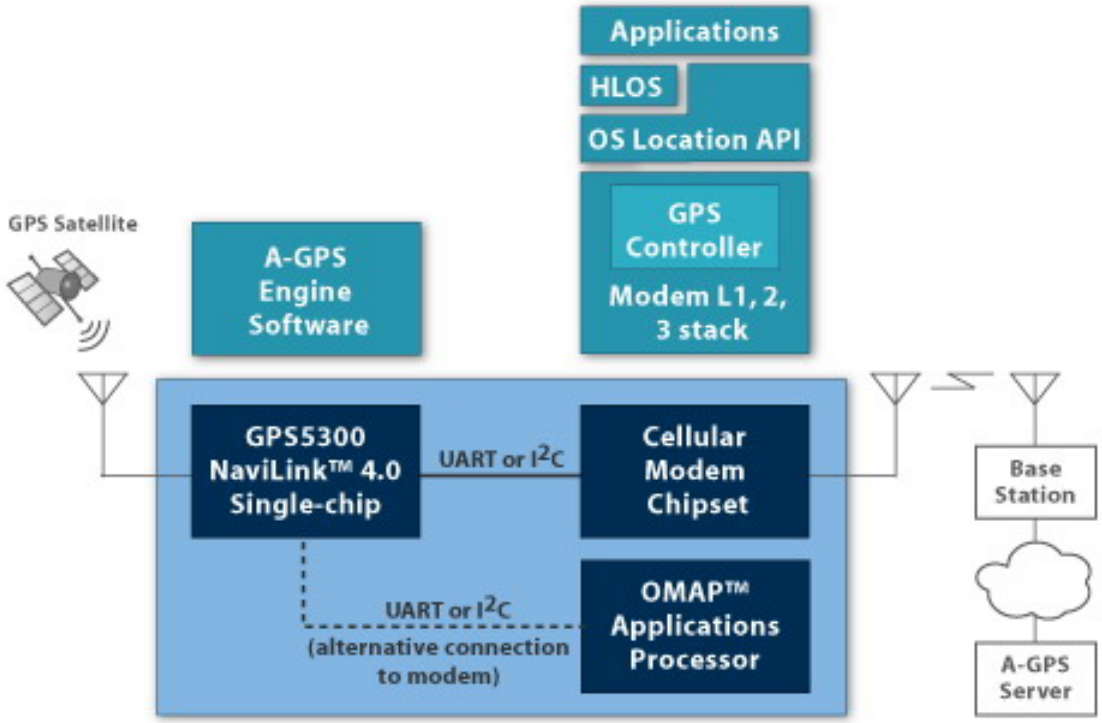
3.7.2 GPS v mobilních telefonech Nokia

GPS modul v mobilních telefonech Nokia z poslední doby je dodáván firmou Texas Instruments. Jedná se o jednočipové řešení optimalizované pro mobilní telefony. Ve většině případech se jedná o NaviLink 4.0 Single-Chip Solution: GPS5300 podporující jak GPS, tak A-GPS funkce, zobrazen na obrázku 3.8. Systém A-GPS je vylepšená verze navigač-



Obrázek 3.8: NaviLink 4.0 GPS5300 [50].

ního systému GPS. Snaží se nahradit chybějící signál v místech, kde není přímá viditelnost na družice. Snaží se umožnit kontakt mobilního přístroje se satelitem nepřímo. Využívá se zde asistenčních zařízení instalovaných do sítě, které zprostředkují komunikaci s družicemi GPS. Technologie A-GPS je založena na IP technologii. Lokalizační server komunikuje



Obrázek 3.9: A-GPS komunikace [51].

přímo s mobilním přístrojem prostřednictvím protokolu vyšší vrstvy IP, který je zaveden do mobilní sítě. Výměna informací mezi telefonem a lokalizačním serverem probíhá přes běžná datová spojení, např. GPRS.

Již zmíněné lokalizační servery jsou schopny vyhodnotit pozici mobilního telefonu přímo na Zemi, provést některé potřebné početní úkony a zjištěné výsledky spojit se signálem satelitu GPS. Tím tedy odpadne problém nedostupnosti satelitního signálu ve stíněných oblastech. Polohu v takovém případě nebude určovat satelit, ale telefon. Odvodí se podle okolních stanic BTS. Tato informace se zpracuje a lokalizační server již zajistí komunikaci se satelitem. Konečné údaje jsou poté poskytovány mobilnímu telefonu opět po bezdrátové mobilní síti. Pomocí technologie A-GPS získává mobilní zařízení data o oběžných drahách, frekvencích a provozních schopnostech satelitů z mobilní sítě, a dokáže tak analyzovat i slabší signály ze satelitů během několika sekund [7].

Princip A-GPS je znázorněn na obrázku 3.9. Světlemodrá část, obsahující GPS5300 čip a Cellular Modem čip, reprezentuje mobilní telefon. V levé části mobilního telefonu je zobrazena reprezentace antény pro GPS a GPS satelit. V pravé části je znázorněna komunikace s GSM stanicí přes GSM anténu a anténu GSM stanice. Nad reprezentací čipů v mobilním telefonu jsou znázorněny vrstvy využívané při GPS komunikaci.

3.7.3 Přístup k GPS v OS Symbian S60

Symbian S60 3rd Edition má zabudovanou podporu pro GPS navigaci. Podporovány jsou jak integrované, tak externí Bluetooth GPS moduly.

Existují dva přístupy k implementaci komunikace s GPS modulem.

- První způsob je využití **nízkoúrovňové komunikace** s externím Bluetooth modulem. Tento přístup byl široce využíván v S60 2nd Edition. Způsoboval veliké množství technických problémů a s některými externími čidly nefungoval korektně. Také nefunguje s interními čidly. Tento přístup se v současné době již nepoužívá.

- Druhý, doporučovaný přístup používá **Location API**, které bylo představeno v S60 2nd Edition Feature Pack 2 a také je podporováno v S60 3rd Edition. Location API obsahuje několik klíčových tříd starajících se o komunikaci s GPS modulem. GPS modul poskytuje informace o aktuálním umístění jako pozici na zeměpisné šířce a délce. Potřebné metody k získání informací o GPS obsahují:

  – Informace o počtu GPS modulů.
  – Informace o aktuálním stavu GPS modulu.
  – Informace o aktuální pozici.

  Z těchto informací se dá určit dostupnost signálu a v případě nalezení dostatečného počtu družic i aktuální poloha [16], [20].

Kapitola 4

Přístupy k analýze dat

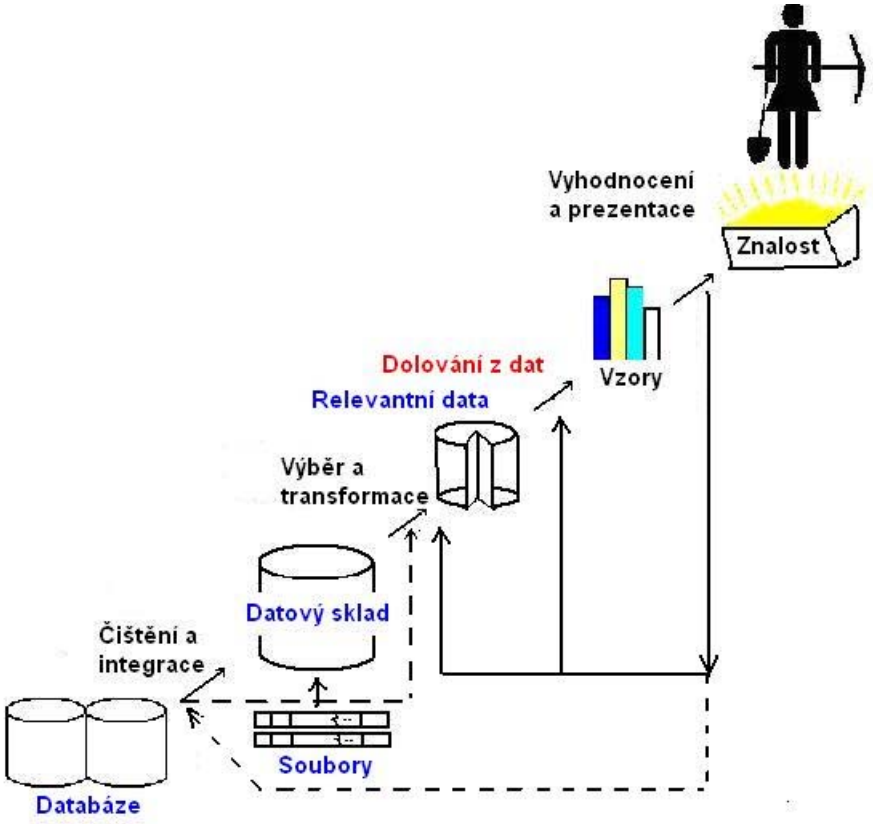
Kapitola popisuje analýzu dat získaných z jednotlivých senzorů, volbu metody k analýze a předpokládané výstupy. Také popisuje klasifikaci dat získaných analýzou senzorových výstupů.

4.1 Získávání znalostí

Získávání znalostí je extrakce zajímavých modelů dat a vzorků z velkých objemů dat. Jedná se o proces sestávající se z řady kroků, které se zpravidla v určitých iteracích opakují (viz. obr. 4.1).

Jedná se o následující kroky:

- Čištění dat - cílem je vypořádání se s chybějícími daty, odstranění šumu a vyřešení nekonzistence dat.
- Integrace dat - cílem je integrace dat pocházejících z několika datových zdrojů. Často se integrace a čištění dat provádí společně. Jednak proto, že vyčištěná data potřebujeme někam ukládat, jednak proto, že jedním ze zdrojů nekonzistence jsou typicky data pocházející z více zdrojů. V takovém případě jsou data jsou data ukládána do datového skladu.
- Výběr dat - cílem je vybrat data, která jsou pro řešení dané analytické úlohy relevantní.
- Transformace dat - cílem je transformovat data do konsolidované podoby vhodné pro dolování. Může jít například o sumarizaci nebo agregaci.
- Dolování dat - jádro procesu získávání znalostí, jehož cílem je aplikací určité metody a konkrétního algoritmu extrahovat z dat vzory, resp. vytvořit model dat.
- Hodnocení modelů a vzorů - cílem je identifikovat skutečně zajímavé vzory pomocí mír užitečnosti.
- Prezentace znalostí - cílem je prezentovat výsledky dolování uživateli využitím technik vizualizace a reprezentace znalostí [57].



Obrázek 4.1: Dolování dat [57].

Data lze dolovat například z relačních databází, databází sekvencí, prostorových databází, textových databází, multimediálních databází, proudů dat a z webu.
Dále se dolování bude zaměřovat na dolování z proudů dat.

Proudy dat - data produkovaná souvisle, mezi jejichž vlastnosti patří především tyto: jejich objem je obrovský, potenciálně nekonečný, dynamicky se mění, umožňuje pro online analýzu zpravidla pouze jeden průchod, dostupný je pouze omezený počet předchozích dat v proudu.

Typy dolovacích úloh můžeme rozdělit do dvou základních skupin:

- Deskriptivní - charakterizují obecné vlastnosti analyzovaných dat.
- Prediktivní - na základě analýzy současných dat provádí dedukci pro předpověď budoucího chování. Představitelem tohoto typu úloh je například klasifikace.
- Úlohy zaměřené na odhalování vztahů mezi atributy. Patří sem dolování frekventovaných vzorů (vzory, které se vyskytují v datech často), asociací a korelací.

Klasifikace - patří mezi prediktivní úlohy dolování dat. Cílem klasifikace je nalézt takový model, který popisuje a současně rozlišuje třídy dat, a ten potom použít pro predikci třídy objektu, jehož zařazení neznáme. Celý proces klasifikace sestává ze tří kroků:

- Trénování (učení) - na základě analýzy tzv. trénovací množiny je vytvořen klasifikační model.

- Testování - hodnocení kvality vytvořeného modelu použitím testovacích dat.
- Aplikace - použití modelu pro klasifikaci objektu, jehož třídu neznáme [57].

Trénovací a testovací data jsou data objektů u nichž známe třídu. V prvním případě tuto informaci využíváme pro vytvoření modelu, ve druhém k otestování modelu, tedy určení četnosti případů, kdy vytvořený model neklasifikuje správně. Na základě výsledku testování buď model upravíme nebo vytvoříme znovu, nebo ho prohlásíme za připravený k použití [57].

4.1.1 Předzpracování dat

Data většinou nejsou v takovém stavu, abychom je mohli bezprostředně postoupit dolovacímu algoritmu k dolování. Obsahují velice často data, která jsou zašuměná, nekonzistentní a některé hodnoty mohou chybět. Taková data označujeme jako „nečistá“ (v angličtině dirty). Nízká kvalita vstupních dat by mohla vést k nepřesným nebo dokonce nesprávným a zavádějícím závěrům vyplývajícím z vydolovaných znalostí.

Existují následující čtyři základní projevy nekvality dat:

- Data jsou nekompletní - mohou chybět hodnoty některých atributů, protože jejich zadání při sběru dat bylo nepovinné a hodnota nebyla zadána nebo došlo k poruše zařízení nebo k chybě programu pro sběr dat.
- Data jsou zašuměná - znamená, že atributy obsahují nesprávné nebo odlehlé hodnoty. Opět může existovat celá řada příčin. Mohlo dojít k poruše zařízení pro sběr dat, k lidské chybě, kterou neodhalila aplikace pro sběr dat, špatné pochopení významu požadovaného údaje apod.
- Data jsou nekonzistentní - potenciálním zdrojem je každá redundance dat. K té může dojít při použití dat z několika datových zdrojů [57].

Ve všech těchto případech je nutné tyto nedostatky dat odstranit. To zajišťují hlavní úlohy předzpracování, mezi které patří:

- Čištění dat - cílem je odstranit chybějící hodnoty, identifikovat a odstranit odlehlé hodnoty a řešit nekonzistence.
- Integrace dat - integrace dat pocházejících z různých datových zdrojů.
- Transformace dat - transformace dat do tvaru vhodného pro řešení dané dolovací úlohy. Patří sem normalizace a agregace.
- Redukce dat - cílem je redukovat objem dat. Patří sem kromě již zmíněné agregace výběr podmnožiny atributů, redukce dimenzionality a redukce počtu hodnot. Kromě agregace musí redukce zachovat charakter původního neredukovaného datového souboru.
- Diskretizace dat - patří také mezi metody redukce dat, ale má pro dolování zvláštní význam. Jde o redukci počtu hodnot atributů. Týká se především numerických (spojitých) dat [57].

Čištění dat je důležité pro správné dolování dat. Z metod čištění dat je pro získávání hodnot ze senzorů důležité odstranění šumu. K jeho odstranění slouží následující techniky:

- Plnění (binning) - tato technika vyhlazuje setříděná numerická data tak, že zohledňuje hodnoty v blízkém okolí. Provádí tedy lokální vyhlazení. Probíhá ve dvou krocích. V prvním se setříděné hodnoty rozdělí do tzv. košů (bins), zpravidla stejné frekvence (equal-frequency), tj. tak, že každý koš obsahuje přibližně stejný počet hodnot. Druhý krok představuje samotné vyhlazování. Hodnoty v koši se nahradí průměrem koše, mediánem koše nebo každá z hodnot koše tou hraniční hodnotou (minimální, resp. maximální hodnota v koši), ke které má daná hodnota blíže.

- Regrese - data se vyhladí tak, že se nahradí hodnotami danými regresní křivkou. V případě lineární regrese se hledá přímka, která nejlépe aproximuje hodnoty dvou atributů. Potom lze použít jeden atribut k predikci (v tomto případě vyhlazení) hodnoty atributu druhého. Vícenásobná lineární regrese je rozšíření lineární regrese na více než dva atributy.

- Shlukování - může být účinnou metodou pro nalezení odlehlých hodnot. Potenciálně to budou ty, které nebudou zařazeny do žádného shluku [2].

4.1.2 Metody pro zpracování proudů dat

Proudy dat je nepraktické procházet ve více průchodech. V některých případech více průchodů nemusí být možné. Je tedy nutné zanalyzovat data v jednom průchodu. Při průběhu analýzy dat ovšem nastává rozpor mezi kvalitou analýzy a velikostí úložného prostoru. Většinou se volí průměr mezi kvalitou a velikostí. Obecně platí, že se vzrůstající velikost úložného prostoru roste i kvalita analýzy.

- **Náhodné vzorkování**
  Metoda spočívá v náhodném výběru několika nezkreslených vzorků bez náhrady. V případě velkého množství dat je velice nákladné získat tento vzorek dat. Tato nevýhoda se dá obejít udržením sady kandidátů ve výběru, kteří se obnovují s pravděpodobností $\frac{s}{N}$. Kde N je počet prvků v proudu a s je velikost vzorku dat.

- **Posuvné okno**
  Používá metodu využití posledních vzorků dat, namísto použití celkového souboru dat. Data jsou platná po dobu $t + w$, kde t je doba vygenerování nového vzorku dat, w je délka okna. Tato metoda využívá malé množství paměti, z důvodu využívání malého okna.

- **Histogram**
  Histogramy dokáží stručně zachytit distribuci hodnot v datové množině. Dělí data do košů (bucket), které mají svoji šířku (rozsah koše) a hloubku (počet položek v daném koši). Kvůli své nepřesné distribuční funkci se využívají jeho upravené verze (např. V-Optional Histogram). V-Optional Histogram aproximuje po částech konstantní funkci f množinou hodnot $v1, \ldots, vn$, tak aby se minimalizovala chyba

- **Multidimenzionální metody**
  K vypořádání se s velkým množstvím dat využívají redukující metody. Využívají hierarchickou strukturu stromů. Pro vytvoření shrnujících dat se používají Wavelety,

jejich koeficienty jsou projekcí daného signálu do ortogonální množiny bázových vektorů. Používají se při počítání multi-dimenzionálních agregací, „data cube“ aproximací, selektivních odhadů, apod [2].

Pro získání výsledného vyhodnocení dat ze senzorů je využita Naivní Bayesovská klasifikace. Klasifikace je popsána v 4.2.

4.2 Naivní Bayesův klasifikátor

Naivní Bayesův klasifikátor je jednoduchý klasifikátor založený na Bayesově teorému s předpokladem nezávislosti mezi příznaky. Tento klasifikátor můžeme s výhodou použít tam, kde jiné klasifikátory (rozhodovací stromy) selhávají kvůli nemožnosti použít velké množství příznaků. Naivní Bayesův klasifikátor může efektivně pracovat i s několika stovkami, až tisíci příznaků. Tento klasifikátor je velmi úspěšný a velmi často se využívá k různým účelům.

V praxi se klasifikátor používá tak, že se podle vzorce stanoví pravděpodobnosti pro všechny možné významy a z nich se vybere význam s největší pravděpodobností. Výhodou naivního Bayesova klasifikátoru je, že mu stačí pouze malé množství trénovacích dat pro odhadnutí potřebných pravděpodobností [9].

4.2.1 Pravděpodobnostní model

Pravděpodobnostní model pro klasifikátor je modelem podmíněným:

$$p(C|F_1, \ldots, F_n), \tag{4.1}$$

kde C je závislá třídní proměnná a $F_1, \ldots, F_n$ jsou příznakové proměnné. Použitím Bayesova teorému můžeme tento vztah zapsat jako:

$$p(C|F_1 \ldots, F_n) = \frac{p(C) \cdot p(F_1, \ldots, F_n|C)}{p(F_1, \ldots, F_n)} \tag{4.2}$$

V praxi nás často nezajímá konkrétní hodnota pravděpodobnosti, ale její relativní poměr k ostatním pravděpodobnostem. Proto můžeme zlomek upravit odstraněním konstanty ve jmenovateli, čímž se výpočet zjednoduší. Upravený vzorec pak bude vypadat následovně:

$$p(C) \cdot p(F_1, \ldots, F_n|C) \tag{4.3}$$

Opakovaným aplikováním definice podmíněné pravděpodobnosti můžeme vzorec upravit na tvar:

$$p(C|F_1, \ldots, F_n) = p(C) \cdot p(F_1, \ldots, F_n|C) = \tag{4.4}$$

$$= p(C) \cdot p(F_1|C) \cdot p(F_2, \ldots, F_n|C, F_1) = \tag{4.5}$$

$$= p(C) \cdot p(F_1|C) \cdot p(F_2|C, F_1) \cdot p(F_3, \ldots, F_n|C, F_1, F_2) \tag{4.6}$$

Nyní uplatníme předpoklad nezávislosti příznakových proměnných, který říká, že každý příznak F_i je podmíněně nezávislý na kterémkoliv jiném příznaku F_j pro $i \neq j$. Tuto podmínku můžeme zapsat následovně:

$$p(F_i|C, F_j) = p(F_i|C) \tag{4.7}$$

Proto můžeme předcházející vzorec přepsat na následující tvar:

$$p(C|F_1,\ldots,F_n) = p(C)\cdot p(F_1|C)\cdot p(F_2|C)\cdot p(F_3|C)\ldots p(C|F_1,\ldots,F_n) = p(C)\prod_{i=1}^{n}\cdot p(F_i|C) \tag{4.8}$$

Můžeme pravděpodobnost toho, že při daných příznacích $F_1, F_2, \ldots, F_n$ bude jev patřit do třídy C, určit následujícím vzorcem:

$$p(C|F_1,\ldots,F_n) = \frac{1}{Z}\cdot P(C)\cdot\prod_{i=1}^{n}\cdot p(F_i|C), \tag{4.9}$$

kde Z vyjadřuje jistou konstantu, která je závislá pouze na nepodmíněných pravděpodobnostech. Ve skutečnosti můžeme Z vyjádřit jako:

$$Z \ = p(F_1,\ldots,F_n) \tag{4.10}$$

Výsledné rozhodování klasifikátoru pak můžeme zapsat vztahem:

$$classify(f_1,\ldots,f_n) = argmax_c P(C=c)\cdot\prod_{i=1}^{n}\cdot p(F_i=f_i|C=c) \tag{4.11}$$

Pro další informace viz. [57], [24] a [8].

4.2.2 Praktické využití

Jelikož výstupy z analyzátorů jsou spojité hodnoty, které jsou diskretizované, je použití Bayesova klasifikátoru vhodné. Klasifikace probíhá na principu příslušnosti vstupních vzorků dat do jednotlivých tříd. Jeli některý z atributů prázdný, znamená to, že pravděpodobnost jeho výskytu v daném vzorku bude nulová. Ve výsledném násobení by tato hodnota způsobila vynulování celkové pravděpodobnosti vzorku. Problém se řeší přidáním malé konstanty μ ke všem příznakovým hodnotám. Tento proces se nazývá Laplaceovo vyhlazování [24].

4.2.3 Praktické řešení určování chování v situacích

Zde je na jednoduchém příkladě reprezentován příklad použití naivního Bayesova klasifikátoru pro určování chování mobilního telefonu na základě vstupních údajů. Množina pravděpodobností chování mobilního telefonu byla určena z několika pokusů a odpovídá reálnému chování uživatelů.

Mějme vzorek dat definovaný následovně:

X = (pohyb="1„; světelnost="1„; hluk="2„; vzdálenost=" >= 200"),

pak pravděpodobnost, že prostředí vyžaduje hlasité vyzvánění s vibračními prvky, určená mým programem je:

P(hlasité vyzvánění s vibracemi = „ano") = 12/48 = 0,25
P(hlasité vyzvánění s vibracemi = „ne") = 36/48 = 0,75

Neboli, do třídy C_1 je klasifikováno 14 vzorků a do třídy C_2 36.

Zjistěme jednotlivé pravděpodobnosti pro každou zanalyzovanou hodnotu:

P(pohyb=„1“|hlasité vyzvánění s vibracemi = „ano“) $= 4/12 \quad = 0,33$
P(pohyb=„1“|hlasité vyzvánění s vibracemi = „ne“) $= 8/36 \quad = 0,22$
P(světelnost=„1“|hlasité vyzvánění s vibracemi = „ano“) $= 2/12 \quad = 0,16$
P(světelnost=„1“|hlasité vyzvánění s vibracemi = „ne“) $= 10/36 \quad = 0,27$
P(hluk=„2“|hlasité vyzvánění s vibracemi = „ano“) $= 10/12 \quad = 0,83$
P(hluk=„2“|hlasité vyzvánění s vibracemi = „ne“) $= 2/36 \quad = 0,05$
P(vzdálenost=“ >= 200“|hlasité vyzvánění s vibracemi = „ano“) $= 5/12 \quad = 0,416$
P(vzdálenost=“ >= 200“|hlasité vyzvánění s vibracemi = „ne“) $= 2/36 \quad = 0,05$

Z těchto pravděpodobností můžeme vypočítat:
$P(X|$hlasité vyzvánění s vibracemi = „ano“$) \quad = 0,33 \cdot 0,16 \cdot 0,83 \cdot 0,416 \quad = 0,018$
$P(X|$hlasité vyzvánění s vibracemi = „ne“$) \quad = 0,22 \cdot 0,27 \cdot 0,05 \cdot 0,05 \quad = 0,00015$

Dále:
$P(X|$hlasité vyzvánění s vibracemi = „ano“$)P($hlasité vyzvánění s vibracemi = „ano“$) =$
$= 0,018 \cdot 0,25 = 0,0045$
$P(X|$hlasité vyzvánění s vibracemi = „ne“$)P($hlasité vyzvánění s vibracemi = „ne“$) =$
$= 0,00015 \cdot 0,75 = 0,0001125$

Maximum tedy nastalo pro třídu C_1, klasifikátor tedy vyhodnotí, že zadané okolní prostředí s větší pravděpodobností bude vyžadovat použití hlasitého vyzvánění s vibracemi.

4.3 Akcelerační data

V současné době není přesnost používaných akcelerometrů velká. Objevuje se rušení na výstupech, které způsobuje i v nehybné pozici akcelerometru generování různých hodnot. Kompenzace těchto výchylek je obtížná a nikdy je nelze úplně eliminovat. S tímto faktem se musí v analýze dat počítat a přizpůsobit tomuto algoritmus.

Nejjednodušším přístupem k eliminaci šumu je využití posuvného okna, do kterého se budou zaznamenávat data. Následně se z dat vypočítá průměr okna, který se použije k dalšímu zpracování. Tento přístup dostatečně eliminuje výchylky a chybovost výstupu se mnohonásobně sníží. Také není výpočetně ani paměťově náročný, proto lze použít v realtime výpočtech i na méně výkonných přístrojích jako jsou mobilní telefony.

Pro odpovídající analýzu dat je nutné odchytit několik vyfiltrovaných výstupů. Jako ideální a dostačující počet hodnot se mi během testování podařilo určit čtyřicet až padesát hodnot. Tento počet hodnot je vygenerován za méně než jednu vteřinu. Při odpočítání prvních pěti záznamů, které jsou málo ovlivněné filtrováním, se dostávají ve většině případu stabilní hodnoty s maximální chybou přibližně dvě jednotky. Při snaze o snížení časové náročnosti, lze snížit počet odchycených jednotek na dvacet. Pod touto hranicí jsou už výsledky méně spolehlivé a tudíž je nelze doporučit k další analýze.

Výsledek analýzy se dosáhne tak, že se zjistí kolik záznamů přesahuje určitý práh. Jestliže počet výkyvů je větších než dva na jednu osu, je předpokládáno, že mobilní telefon je v pohybu a výsledek analýzy je ukončen. Pokud počet výkyvů je menší, je předpokládáno, že mobil je v klidovém stavu.

Počet výkyvů byl nastaven na tři z důvodu občasných odchylek u senzorových dat. Hodnota tři byla získána několika pokusy a byla shledána dostačující.

Výstupem analýzy je informace o pohybu nebo o klidu.

4.4 Data ze světelného senzoru

Jak bylo již zmíněno v kapitole 3.4.2, lze použít světelné čidlo pouze ve verzích Symbianu od S60 3rd FP2 a vyšší. I když je aplikace napsaná a odladěná na Symbianu S60 3rd FP1, obsahuje modul na použití světelného čidla, které nahradí nutnost využívat fotografický senzor (viz. 4.5).

Světelný senzor pracuje na obdobném přístupu jako senzor akcelerační. Senzor vyhodnotí okolní podmínky a vrátí hodnotu odpovídající intenzitě světla v okolí. Ale na rozdíl od akceleračního čidla, zde není potřeba získávat více informací. Ve většině případů je jedna hodnota postačující. Podle vrácené hodnoty se výstup zanalyzuje, rozdělí se na tři hodnoty definující tmu, šero a světlo, a následně se odešlou pro další analýzu.

4.5 Data z fotografického senzoru

Pokud není mobilní telefon vybaven světelným čidlem, případně k němu nemá přístup, musí se použít jiný postup k získání světelných informací z okolního prostředí. Jako ideální náhradní řešení je využití fotografického senzoru.

Nutnou podmínkou je, že senzor nesmí být využíván. Je k němu totiž potřebný výhradní přístup. Pokud je tato podmínka splněna, může se začít s analýzou. Po výběru senzoru, který se provádí automaticky podle dostupných senzorů v mobilním telefonu, se pořídí fotografie. Uloží se do paměťového prostoru a spustí se analýza. Kvůli co nejmenšímu vytížení mobilního telefonu je zvoleno nejmenší přípustné rozlišení. Ve většině případů odpovídá 352 x 288 pixelů. Použit je nativní formát OS Symbian. Jde o bitmapu spravovanou bitmapovým serverem. Tento formát byl zvolen kvůli jednodušší prezentaci, ale hlavně kvůli univerzálnosti. Při použití jiné reprezentace fotografie byl problém s vytvořením univerzálního algoritmu pro použití čelní i zadní kamery, kvůli jiným možnostem jednotlivých senzorů.

Analýza světelnosti je s použitým formátem jednoduchá. Návratová hodnota jednotlivého pixelu odpovídá jeho intenzitě světla, kdy nejtmavější má hodnotu 0 a nejsvětlejší odpovídá hodnotě 4080. Proto je nejjednodušší vypočítání průměru hodnot v jednotlivých pixelech. Výsledná hodnota pak odpovídá intenzitě světla v okolí a může se tedy použít pro další zpracování.

4.6 Zvuková data

Pro kompletní analýzu okolních vlivů je potřeba znát také okolní hluk. Dá se také říct, že tento faktor má na vliv mobilních komunikací největší vliv. V případě této diplomové práce se jedná o slyšitelnost vyzvánění, případně o jeho ztišení. Bylo by nepříjemné, kdyby nebylo vyzvánění slyšitelné kvůli špatně zanalyzované okolní hladině zvuku, nebo bylo příliš hlasité.

Pro analýzu zvuku je nutné mít výhradní přístup k nahrávacím zdrojům mobilního telefonu. V případě, že je tento přístup umožněn, dojde k nahrání několika vzorků zvuku v nekomprimovaném formátu do paměti mobilního telefonu. Pokud je nahrávání úspěšné, spustí se analýza zvuku. Z jednotlivých pokusů vyšlo najevo, že je výhodné používat pouze druhou polovinu nahraných vzorků dat. Je to z důvodu, že první část obsahuje inicializační

data streamu. Kvůli malé velikosti nahraných dat, by bylo hledání začátku zvukových informací zbytečně časově náročné. I přes ztrátu poloviny vzorků dat je výsledek dostačující a v porovnání s analýzou celého vzorku dat dokonce i několikanásobně přesnější.

Pro analýzu vzorků zvuku byla zvolen Rychlá Fourierova transformace[1] (dále jen *FFT*), speciálně šlo o metodu sloučeného algoritmu. FFT je věnována kapitola 4.7. Výhoda této metody je v její efektivnosti a rychlosti analýzy. Výstupem FFT je vypočítané spektrum vstupního signálu, z kterého se následně jednoduše vypočítá energie zvuku. Z energie zvuku se následně určí výsledná hladina zvuku v okolí. Návratová hodnota je jedna ze tří intenzit: ticho, normální hlasitost, hluk. Tato hodnota je dále zpracována pro výslednou analýzu prostředí.

4.7 Rychlá Fourierova transformace

Rychlá Fourierova transformace (dále také *FFT*) je efektivní algoritmus pro spočtení diskrétní Fourierovy transformace[2] (dále jen *DFT*) a její inverze. FFT je velmi důležitá v mnoha oblastech, od digitálního zpracování signálu a řešení parciálních diferenciálních rovnic až po rychlé násobení velkých celých čísel.

Při výpočtu DFT se vlastně jedná o ekvivalentní vyčíslení hodnoty polynomu stupně $N-1$ s koeficienty x_n v bodě $e^{-\frac{2\pi i}{N}}$. Optimálním postupem pro výpočet polynomů je Hornerovo schéma, které potřebuje $N-1$ násobení a sčítání, což pro celou transformaci dává počet operací $2 \times N \times (N-1)$, tedy přibližně N^2 operací. Už tedy i pro malý počet vzorků čas výpočtu neúměrně narůstá. Proto je nutné využívat jiné časově úsporné metody, tedy takové, které využívá například FFT.

Nechť $x_0, ..., x_{N-1}$ je komplexní číslo. DFT je definováno vzorcem 4.12

$$F_k = \sum_{n=0}^{N-1} x_n (e^{-\frac{2\pi i}{N}})^{nk} \quad k = 0, \ldots, N-1. \tag{4.12}$$

Přímé vyhodnocení těchto sum by zabralo $O(N^2)$ aritmetických operací. FFT naproti tomu poskytuje složitost pouze $O(NlogN)$ operací. Obecně jsou FFT algoritmy založeny na faktorizaci N, nicméně existují i FFT algoritmy se složitostí $O(NlogN)$ pro všechna N, tedy i pro prvočísla [44].

4.7.1 Cooley-Tukey algoritmus

Cooley-Tukey algoritmus je nejpoužívanější variantou FFT algoritmu. Byl prezentován v práci J. W. Cooley a J. W. Tukey v roce 1965 [35]. Je příkladem rozděl a panuj algoritmu, který rekurzivně dělí DFT s velikostí složeného čísla do menších DFT o velikostech N_1 a N_2. Díky tomu a znalosti definice DFT, lze přepsat vzorec do tvaru použitelného pro FFT:

$$F_k = \hat{F}_k + (e^{-\frac{2\pi i}{N}})^k \check{F}_k, F_{k+\frac{N}{2}} = \hat{F}_k - (e^{-\frac{2\pi i}{N}})^k \check{F}_k \quad k = 0, \ldots, (\frac{N}{2}) - 1. \tag{4.13}$$

Nejpoužívanější podobou Cooley-Tukey algoritmu je dělení transformace v každém kroku na dva díly o velikosti $fracN2$ (čímž je transformace omezena na velikosti mocniny dvou), nicméně je možné použít kteroukoli faktorizaci. Přestože je idea rekurzivní,

[1] v originále „Fast Fourier transform“
[2] v originále „Discrete Fourier transform“

většina tradičních implementací algoritmus modifikuje, aby se vyhnuly explicitní rekurzi. Vzhledem k tomu, že Cooley-Tukey algoritmus dělí DFT do několika menších DFT, je možné zkombinovat tento algoritmus s kterýmkoliv jiným DFT [35].

4.7.2 Další algoritmy pro výpočet FFT

Cooley-Tukey algoritmus není jediný zastánce FFT. Za zmínku jistě stojí i následující algoritmy:

- Algoritmus s redukcí kmitočtu spočívá v rozdělení posloupnosti vzorků $\{x_n\}_0^{N-1}$ na dvě poloviny s přirozeným uspořádáním vzorků $\{x_n\}_0^{(\frac{N}{2})-1}$, $\{x_n\}_{(\frac{N}{2})-1}^{N-1}$. Provede-li se definiční transformace, uvidí se, že je možno výstupní posloupnost rozdělit na sudou a lichou část, kde sudá část je dána transformací N/2-členné součtové posloupnosti vstupních dvou posloupností a lichá část je dána transformací rozdílové posloupnosti. Časová složitost je i zde $O(NlogN)$ [35].

- Sloučený algoritmus je založen na vlastnosti transformací reálných posloupností, na její redudantnosti: $F_k = F_{N-k}^*$. Při provádění transformace se zároveň vypočítávají Fourierovy koeficienty pro dvě nezávislé posloupnosti x_n, y_n tak, že se spočítá transformace posloupnosti $x_n + ig_n$. Z těchto hodnot se pak příslušné koeficienty určí jako lineární kombinace reálné a imaginární části [35].

- Algoritmus FFT s prvočíselným rozkladem je algoritmus, který je v některých případech rychlejší než Cooley-Tukey algoritmus. Postup je založen na rozdělení transformace N-složkového vektoru, kde $N = N_1 N_2 \dots N_n$ a N_i jsou prvočísla či jejich mocniny, na n transformací vektorů o N_i složkách [35].

Kapitola 5

Návrh aplikace

Tato kapitola se věnuje návrhu aplikace pomocí diagramů jazyka UML. Jedná se o jazyk, který je v současné době hojně využíván v softwarovém inženýrství. Hlavní výhoda jazyka spočívá v jeho grafické reprezentaci, která se velice dobře hodí pro vizualizaci, specifikaci a návrh softwarových projektů. UML podporuje objektově orientovaný přístup k návrhovým prostředkům, což je velice výhodné, protože většina současných aplikací je implementována pomocí objektově orientovaných programovacích jazyků. Využitím jazyka UML je možné rychle a intuitivně přecházet z návrhu do implementace aplikace samotné [43].

5.1 Konkurenční aplikace

Před provedením analýzy a návrhu programu je dobré ověřit stav konkurenčních aplikací.

Za zmínku určitě stojí aplikace *Auto Profiles 2.0*[1] vyvinutá společností *SymbianGuru*. Tato aplikace je založena na časovém plánovači, který v zadaný čas změní aktuální nastavení profilu. Také dokáže změnit přímo položky v profilu a tím profil přizpůsobit aktuální potřebě. Tento program ovšem nevyužívá okolní informace k přizpůsobení svého schování.

Další z řady programů upravující vyzvánění je *Noise Detection*[2] pro platformu Windows Mobile. Program dokáže na základě okolní intenzity zvuků přizpůsobit hlasitost vyzvánění. Bohužel nedokáže využívat více senzorů pro svoji analýzu.

5.2 Prvotní analýza a plán návrhu

Jeden z prvních kroků při tvorbě aplikací je specifikace neformálních požadavků na vyvíjený software. Jde o zadání požadavků uživatele na funkčnost programu. Jedná se o textovou formu popisu. Na základě této specifikace se vytváří prvotní analýza aplikace.

5.2.1 Neformální specifikace

Cílem vyvíjené aplikace je přizpůsobení vyzvánění mobilního telefonu při příchozím telefonním hovoru podle okolního prostředí. Jedná se o aplikaci s grafickým uživatelským rozhraním, pro operační systém Symbian S60. Uživatel má možnost upravovat určité vlastnosti chování programu a přiřazovat je k určitým telefonním číslům. Také je nutné zachovávat informace o nastavení, aby se program nemusel při každém zapnutí znovu nastavovat.

[1] http://www.symbianguru.com/auto-profiles-2-0-for-series-60-v3-profile-scheduler-including-a-bluetooth
html

[2] http://msdn.microsoft.com/en-us/magazine/cc163341.aspx

5.2.2 Prvotní analýza požadavků

Na základě neformální specifikace aplikace lze odvodit, že se jedná o aplikaci s grafickým uživatelským rozhraním. Budou dostačující implicitní grafické prvky jako je obrazovka s prvky pro nastavení a obrazovka se seznamem.

Část obsluhující hardwarovou část bude nejvýhodnější naimplementovat formou modulů. Moduly budou tvořit výkonnou část aplikace. Každý modul bude, s jistými úpravami, schopný samostatného spuštění. Tím se zajistí plná modularita programu. Ke komunikaci mezi moduly bude využito callbacků. Podle nejpoužívanější třívrstvé architektury budou odděleny jednotlivé vrstvy(GUI, kontroler a vrstva obsluhující hardware přístroje). Bude snaha o co největší využití optimalizovaných metod pro přístup k senzorům.

Ukládání nastavení bude prováděno pomocí vestavěných metod pro ukládání konfiguračních souborů. Bude využita perzistentní paměť mobilního telefonu, aby nedošlo ke ztrátě dat i v případě náhlého odpojení napájení paměti. Metody ukládání si zajišťují vlastní způsob uložení dat a tím odpadne starost o integritu souboru, protože se o ni postará sám Symbian S60.

5.2.3 Plánování projektu

Vývoj aplikace bude prováděn v několika etapách. Jako první budou vytvořeny moduly pro ovládání senzorů, tak aby mohly fungovat nezávisle na sobě. Po jejich optimalizaci a otestování bude první fáze ukončena.

V druhé fázi se začne vytvářet kontroler, který bude ovládat veškeré moduly obsluhující senzory. Dále se v kontroleru vytvoří logika pro komunikaci s modulem zpracovávající upravené hodnoty ze senzorů. Dohromady s vývoje kontroleru se začne vyvíjet modul pro vyhodnocení okolního prostředí.

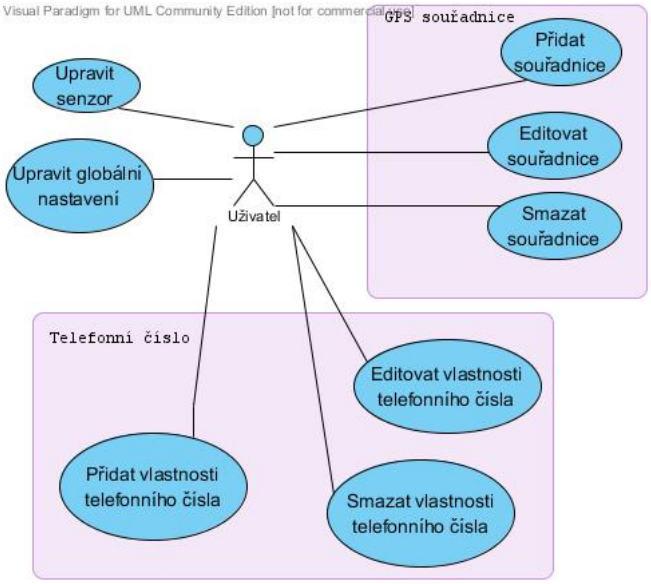
Následovat bude fáze přizpůsobení vyzvánění výstupům z analýzy. To znamená detekce příchozího hovoru, aktivování vyzvánění a jeho včasné ukončení.

V poslední fázi dojde k vytvoření GUI a optimalizaci celého programu s odstraňováním případných chyb.

5.3 Diagram případů užití

Diagramem případů užití, viz. obrázek 5.1 se dá nazvat přehled všech úkonů, které mohou aktéři s aplikací provádět. Je to pohled koncového zákazníka, neboli uživatele, na to, co by aplikace měla ve své finální podobě umět.

Diagram je rozdělen do čtyř oblastí, které reprezentují jednu obrazovku a úkony v něm.



Obrázek 5.1: Diagram případů užití

5.4 Diagram tříd

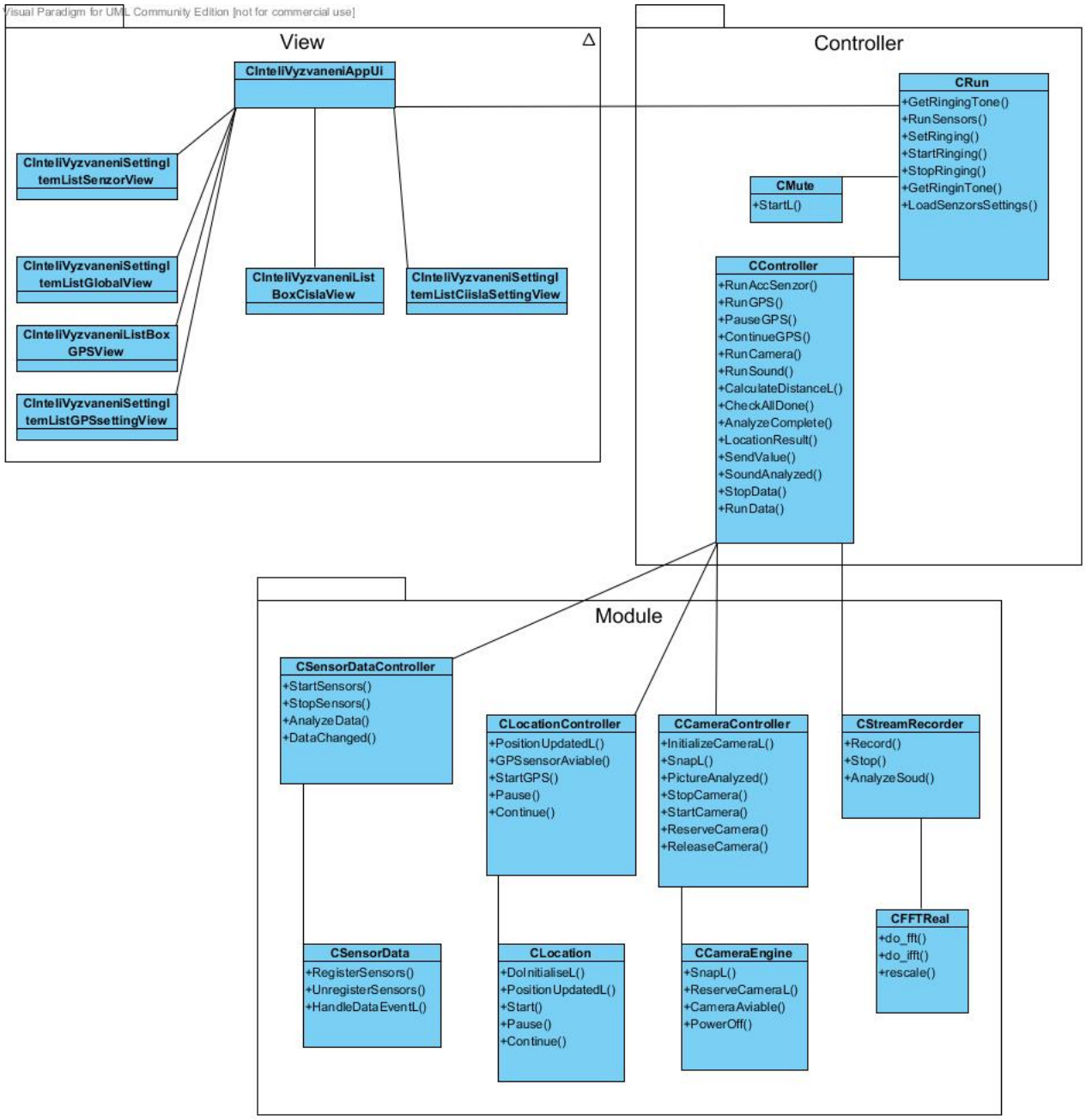
Diagram tříd zobrazuje statické struktury systému pomocí tříd a vztahů mezi nimi. Diagram 5.2 zobrazuje pouze konceptuální model. Nelze jej brát jako závazný, slouží spíš jako návrh z hlediska koncepce aplikace. Jeho úkolem je snaha přiblížit funkce a vlastnosti konkrétních tříd.

Aplikace se bude skládat z několika tříd obsluhujících grafickou část, tříd starajících se o zpracování informací ze senzorů a tříd působících jako logika aplikace, starajících se o správnou interpretaci dat a vyzvánění.

Třídy obsluhující grafickou část budou vytvořeny pomocí nástroje UI Designer. Jedná se o třídy v balíku View.

Třída CRun bude odpovídat za vyzvánění společně se třídou CMute. Bude obsahovat metody pro ovládání vyzvánění, například StartRinging nebo StopRinging a metodu pro načtení nastavení senzorů LoadSenzorsSettings. O analýzu dat ze senzorů se bude starat třída CController přes metody spouštějící a zastavující jednotlivé senzory, např. RunCamera nebo StopCamera. Dále třída CController bude vytvářet instance tříd ovládající senzory.

Třídy v balíku Module se budou starat o správné chování senzorů a o informování nadřazených tříd o ukončení odchytávání informací. Pro každý senzor bude vytvořena třída obsluhující hardware senzoru (např. CSensorDataController) a třída starající se o správnou inicializaci a nastavení senzoru (např. CSensorData). Kontrolery senzorů budou obsahovat metody na spuštění senzoru (např. StartSensors), zastavení senzorů (např. StopSensors), analýzu výstupních dat (např. AnalyzeData) a metody na informování nadřazené třídy o ukončení získávání dat (např. DataChanged). Třídy obsluhující hardware senzorů budou obsahovat metody na zaregistrování senzorů, případně inicializaci, (např. RegisterSensors), jejich odregistrování (např. UnregisterSensors) a metody reagující na změnu dat v senzoru (např. HandleDataEventL). Jako vzorové třídy je použita obsluha akceleračního senzoru.



Obrázek 5.2: Diagram tříd

Kapitola 6

Implementace aplikace

Na počátku implementace se stálo před rozhodnutím, jaké technologie a programovací jazyk zvolit. Kvůli přístupu k senzorům a rychlosti jazyka byl zvolen jazyk C++. A jaké k tomu byly důvody?

6.1 Programovací jazyky na platformě Symbian S60

Jelikož cílová platforma je Symbian OS, který je založený na objektově-orientovaném návrhu (dále *OO*), implementovaném v jazyce C++ (ovšem také na nejnižší úrovni využívá asembleru a jazyka C), bylo rozhodování omezeno na OO jazyky. Další omezením byla podpora jazyka samotným systémem Symbian a schopnost přistupovat k integrovaným senzorům. Pro splnění těchto podmínek byl výběr zúžen na tři programovací jazyky. Jednalo se o jazyk C++, Java a Python.

- **Python pro S60**
  Python pro S60, nazýván také PyS60, je dynamický objektově orientovaný programovací jazyk. Jedná se o port stolní verze Pythonu pro platformu S60. Narozdíl od verze pro stolní počítače, PyS60 obsahuje rozšíření umožňující přístup k velkému množství funkcí mobilního telefonu. Existuje více druhů Pythonovských portů pro mobilní telefony, ty ale nejsou optimalizované pro využití v Symbian OS S60. PyS60 je vyvíjen firmami Apache 2 a Python jako opensource a je založen na Pythonu 2.5.4. Dovoluje přistupovat k většině hardwarových součástí telefonu, jako je fotoaparát, mikrofon, reproduktory, atd. Také dokáže obsluhovat dotykovou obrazovku. Výhodou PyS60 je možnost integrování do ostatních programovacích jazyků [39].

- **Java pro S60**
  Java Platform Micro Edition neboli Java ME[1] je soubor technologií a specifikací pro tvorbu aplikací na mobilní zařízení používající technologii Java. Java ME je také optimalizována pro Symbian S60 a S40. Pro programování Java ME aplikací se většinou používá MIDP architektura. MIDP je aplikační rozhraní J2ME, které definuje, jakým způsobem softwarové aplikace spolupracují s mobilními telefony a pagery. Aplikace, která je vytvořena podle tohoto standardu, se nazývá MIDlet. MIDlet je aplikace podobná apletu. Musí rozšiřovat speciální třídu MIDlet a v telefonu běží z bezpečnostních důvodů v tzv. sandboxu, které nemůže opustit.

[1] dříve známé jako J2ME

Java aplikace je možné provozovat na systémech i s malým množstvím paměti. Program je upraven příslušným JVM. Podle množství paměti se dělí výsledná konfigurace souboru na:

- Connected Limited Device Configuration (CLDC) - jsou to programy běžící na systémech s méně než 512 kB paměti pro prostředí JVM[2]. Z programu jsou odstraněny uživatelské implementace načítání tříd a některé třídy pro verifikaci. JVM se u této konfigurace jmenuje KVM (K-VirtualMachine).
- Connected Device Configuration (CDC) - obsahuje programy běžící na systémech s více než 512 kB paměti pro JVM. Implementují vlastní vlákna, ta mohou být samozřejmě podobná systémovým. Garbage collector je zde vlastní program běžící v CVM. JVM je pojmenována CVM (C-VirtualMachine).

Java programy nemohou přistupovat přímo ke všem hardwarovým součástem telefonu. Pokud je nutné přistoupit k nepřístupným součástem, používá se jako mezistupeň server napsaný v PyS60. Komunikace mezi programy v Java ME a PyS60 probíhá pomocí socketů. Tedy Pythonovský program zastupuje funkci serveru. Javovský program si data přebere a využívá je ke svému účelu [21].

- **C++ pro S60**
  Symbian C++ je přirozený programovací jazyk pro zařízení pracující pod Symbian OS, podobný standardu C++. Od standardního C++ se Symbian C++ liší hlavně v práci s pamětí. V mobilních telefonech je totiž nutné používanou paměť co nejlépe využívat a co nejdříve uvolňovat. Tím, že celý systém je programován v tomto programovacím jazyku, Symbian C++ API poskytuje vývojáři ohromné množství variant přístupu téměř k jakékoliv části mobilního telefonu. Také obsahuje nástroje na tvorbu grafického uživatelského rozhraní. Je považován za nejrychlejší a nejvýhodnější programovací prostředek pro tvorbu programů pod Symbian OS. Jeho nevýhodou je však značná složitost oproti předchozím dvou jazykům [23].

  Jak bylo zmíněno, Symbian C++ má oproti standardnímu ANSI C++ jisté rozdíly:

  - mechanizmus vyjímek - V Symbian C++ je upřednostňováno zachytávání vyjímek pomocí makra TRAP místo mechanizmu trow/catch ve standardním C++. Je to speciální přístup vyvinutý společností Symbian za využití mechanizmu zvaného „leaving“. Toto zachytávání neumožňuje odchytit závažné chyby typu panika (panic), které vedou k ukončení procesu (případně vlákna nebo aplikace). Princip vyjímek je následovný. Výjimka je generována metodou Leave a odchycena pomocí makra TRAP. Metodě Leave je předán celočíselný kód chyby, který může být při odchycení výjimky makrem TRAP načten a zpracován. V tomto je mechanismus úspornější, protože výjimka není jako v C++ parametrizována libovolným objektem.
  - úklidový zásobník (cleanup stack) - Je používán jako prevence před úniky paměti (memory leaks). Jsou zde ukládány ukazatele na objekty, které je následně při ukončení nutné uvolnit.
  - dvoufázová konstrukce - Jedná se o bezpečné vytvoření objektu s využitím úklidového zásobníku.

[2] Java Virtual Machine

– značení datových typů - Speciální značení datových upřesňuje jejich uložení a typ. Jde také o zabezpečení kódu proti překladu nesprávným kompilátorem. Např. ETrue (enum), HBuff (heap).

– řetězce - Místo klasických C++ řetězců jsou používány deskriptory. Obsahují informace o své aktuální a maximální délce [23].

Další informace o rozdílech mezi Symbian C++ a ANSI C++ lze získat na [23].

6.2 Výběr programovacího jazyka

Faktorem pro výběr programovacího jazyka byla jeho rychlost. I když většina algoritmů nevyžaduje speciálně výkonný systém, v případě cílové platformy jsou časové nároky na jednotlivé programovací jazyky velké. Jazyk Java, kvůli nutnosti využívat virtuální stroj, má větší nároky na výkon procesoru. Je to z důvodu, že patří mezi jazyky, které nejsou kompilovány do strojového kódu, nýbrž do tzv. mezikódu, který je poté interpretován virtuálním strojem.

Další podmínkou pro výběr byla možnost ovlivňovat hlasové a vyzváněcí funkce mobilního telefonu. V této podmínce obstál nejlépe programovací jazyk C++. Ostatní mají totiž omezené přístupové možnosti k využívání zdrojů, nebo díky jejich bezpečnostní politice, je nelze provozovat na pozadí bez toho, aniž by musel uživatel do jejich chodu zasahovat.

Také bylo nutné zajistit, aby program fungoval na všech mobilních telefonech, obsahující stejnou verzi OS Symbian, bez nutnosti doinstalovávat další aplikace. Tento přístup porušuje jazyk Python, který ke komunikaci se senzory vyžaduje nainstalování Pythonovského modulu a skcript shellu na cílový mobilní telefon.

Po zhodnocení všech těchto faktů byl výběr programovacího jazyka zřejmý. Byl zvolen objektově orientovaný jazyk C++.

6.3 Výběr vývojového prostředí

Jelikož pro jazyk C++ existují dvě vývojová prostředí, jedná se o Carbide.c++ IDE[3] a Qt Creator IDE, bylo nutné zvolit jedno z nich. Obě prostředí dovolují využít možnosti ladění aplikací, ať na emulátorech nebo na koncovém zařízení (tzv. on-device debugging). Dále obsahují průvodce pro vytváření projektů, tříd, které usnadňují vývoj a starají se o vytvoření základní kostry daných součástí projektu. Také obsahují editory uživatelského rozhraní (UI designer). Využitím editoru uživatelského rozhraní se velice ulehčí práce na tvorbě uživatelského rozhraní aplikace, kdy se vzhled vytváří vizuální formou, intuitivně pomocí přetahování komponent UI myší. Tímto způsobem se dá definovat obsah hlavní obrazovky aplikace, tak také nabídek, sdělení, dialogů a voleb v ovládacích polích. Společně s úpravami se také generuje odpovídající kód. V editoru UI se dá ovšem i přiřadit jednotlivých prvkům obrazovky odpovídající programové funkce a tím se celý program oživí.

- **Carbide.c++ IDE** je nástroj pro univerzální vývoj aplikací jakéhokoliv rozsahu. Je postaven na populárním vývojovém prostředí Eclipse IDE a jeho vlastnostech pro vývoj C/C++ aplikací. Carbide.c++ dovoluje efektivní vývoj aplikací v Symbian C++ a Qt.

[3] Integrated Development Environment

- **Qt Creator IDE** je vývojové prostředí primárně určeno pro využití Qt frameworku založeném na knihovnách C++. Qt je multiplatformní aplikace a UI framework sloužící také pro vývoj web-enabled aplikací[4]. Výhoda je, že takto vzniklý kód je použitelný na více platformách, ať se jedná o stolní počítače, mobilní telefony nebo vestavěné operační systémy, bez jakékoliv úpravy. Jeho použití je ale spíše v dotykově ovládaných telefonech, než v jejich nedotykových verzích, kde je ovládání značně omezené. V době vývoje této diplomové práce bylo obtížné komunikovat s vestavěnými čidly pomocí Qt frameworku.

Po analýze pozitivních i negativních vlastností jednotlivých prostředí jsem se rozhodl využít vývojové prostředí Carbide.c++ a jazyka C++.

6.4 Požadavky na SDK

Jelikož program je vyvíjen na přístroji Nokia N82, bylo zvoleno SDK 3rd Edition Feature Pack 1 [5], které odpovídá tomuto mobilnímu telefonu.

6.5 Implementace obsluhy senzorů

V následujících sekcích je popsána implementace obsluhy senzorů, získávání dat, jejich použití a nalezené komplikace. Každá sekce se zabývá jediným senzorem, respektive analýzou dat.

6.5.1 Akcelerační čidlo

K vývoji algoritmů akceleračního čidla zaznamenávající pohyb (viz. 3.4) bylo nutné rozšířit základní sadu SDK sadou obsahující API pro přístup k senzoru. Tento plugin je volně ke stažení na stránkách firmy Nokia[6]. Bohužel toto SDK neobsahuje plugin pro ladění na emulátoru, je tedy nutné při každém ladění využívat ladění na mobilním telefonu. Program využívající akceleračního čidla může být self-signed.

Pro implementování obsluhy akceleračního senzoru je nutné využít callbacků obsahující informace o změně údajů. Tyto callbacky obsluhuje abstraktní třída MRRSensorDataListener odkazovaná pomocí hlavičkového souboru *RRSensorApi.h*

Nejprve je nutné zaregistrovat posluchače senzoru, k tomu slouží třída CRRSensorApi, které se předá Id senzoru. Až po té je možné přistoupit k odposlouchávání změn. K odposlouchávání je určena metoda HandleDataEventL, přes její parametry se získají informace o senzoru (v případě že je jich zaregistrovaných víc, je nutné odchytávat správné informace podle Id senzoru). Pod odchytu informace se provede její vyfiltrování, k získání stabilní hodnoty bez velkých výkyvů. I v případě malého pohybu je dat s požadovanými odchylkami více, tudíž filtr vrátí správné hodnoty. Po obdržení informací ze senzorů se data uloží do nosné třídy CSensorDataHolder a jsou dále zpracována.

[4] aplikace s podporou internetových funkcí

[5] http://www.forum.nokia.com/info/sw.nokia.com/id/ee46217f-f72a-44b6-8361-833ff782b4a7/s60_3rd_fp1_sdk_f_ReleaseNotes.txt.html

[6] http://www.forum.nokia.com/info/sw.nokia.com/id/4284ae69-d37a-4319-bdf0-d4acdab39700/Sensor_plugin_S60_3rd_ed.exe.html

Jak už bylo vícekrát zmíněno, je nutné data vyfiltrovat, aby se odstranil šum, který je v následné analýze nežádaný. K filtrování slouží třída CSensorDataFilter. V ní je implementován kruhový buffer, obsahující posledních několik hodnot, z kterých se vypočte průměr a odešlou se k dalšímu zpracování. Implementace je dostačující, rychlá a výsledná data neobsahují nežádoucí šum. Jediná nevýhoda je v nemožnosti použití prvních pěti vzorků kvůli malé efektivitě filtru před naplněním bufferu.

Třída, která se stará o celý proces získávání a analýzy dat, se nazývá CSensorDataController. Tato třída zapouzdřuje předchozí třídy a je zodpovědná za vytvoření jejich instancí a jejich správné ukončení. Díky callbacku implementovaným v abstraktní třídě MCSensorDataNotify získá informace o ukončení sběru dat a započne analýzu.

Analýza dat je řešena výpočtem absolutní hodnoty rozdílu mezi jednotlivými vzorky. Tato hodnota se pak dále zpracovává.

Po ukončení analýzy se zašle zpráva nadřazené třídě s výsledkem své analýzy. Toto odeslání se děje také pomocí callbacku tentokrát obsaženého v abstraktní třídě CSensorAnalyzeComplete.

6.5.2 Světelné čidlo

Světelné čidlo využívá vlastností S60 3rd Edition FP2 a tou je jednodušší přístup ke všem zabudovaným senzorům. Obdobně jako akcelerační senzor využívá k získávání informací o stavu čidla callback, tentokráte obsažený v abstraktní třídě MSensrvDataListener. O vytvoření instance třídy se stará metoda StartSensor. Vytvoří kanál na získání informací o obsažených senzorech a z nich vybere ty potřebné. Poté zaregistruje posluchače světelného senzoru pomocí instance třídy CSensrvChannel s parametrem obsahující informaci o senzoru.

Jakmile čidlo získá hodnotu okolního světla, informuje pomocí callbacku třídu zaregistrovanou na odposlechu. Návratová hodnota obsahuje informaci o okolní intenzitě světla definované ve třídě TSensrvAmbientLightData. Data se následně zanalyzují a odešlou se nadřazenému kontroleru.

6.5.3 Fotografický senzor

Fotografický senzor (dále i kamera) je využit tam, kde není dostupný světelný senzor. Implementovaná obsluha fotografického senzoru je složena ze dvou tříd. Jedna se stará o obsluhu kamery na hardwarové úrovni, druhá o správné pořízení fotografie a nastavení kamery.

Třída CCameraEngine obsluhující fotografický senzor se stará o zinicializování a použití kamery. Ke své funkci musí dědit z mixin[7] třídy MCameraObserver. Tato třída v sobě obsahuje callbacky vracející informace o postupu vytváření fotografie od první inicializace, až po oznámení chyb. Pro zdědění z této třídy je nutné mít přítomný hlavičkový soubor *ecam.h*

Jedna z metod ve třídě CCameraEngine stojící za zmínku je její konstruktor druhé fáze ConstructL. Zde je proveden test na přítomnost kamery pomocí metody CamerasAvailable třídy CCamera. Následně je zvolena čelní kamera, pokud je tedy přítomna, v opačném případě se použije hlavní kamera, a je vytvořena její instance.

Pro použití kamery je nutné ji nejdříve zarezervovat pomocí metody Reserve třídy CCamera. Tím se dosáhne jedinečného přístupu k této kameře. O výsledku zarezervování infor-

[7] souvisí s postupem, kdy je objekt „smíchán“ z více jiných objektů (či tříd) tak, že je podtypem každého z nich (např. pomocí vícenásobné dědičnosti)

muje callback ReserveComplete. Pokud zarezervování proběhne úspěšně, kamera se zapne metodou PowerOn. Po zapnutí kameru je zavolán callback PowerOnComplete starající se o informování nadřazené třídy o výsledku zapnutí.

Po úspěšném zapnutí se zjistí podporované formáty a rozlišení. Vybere se ten nejméně paměťově náročný. Následně se alokuje paměť a uloží nastavení do kamery pomocí metody PrepareImageCaptureL. Za pořízení snímku je odpovědná metoda CaptureImage volající callback ImageReady po svém ukončení. Metoda ImageReady je volána s parametry charakterizující fotografii. Jedná se buď o ukazatel na bitmapu uloženou ve formátu CFbsBitmap nebo o ukazatel na pole obsahující komprimovanou fotografii. V případě této aplikace je zvolen formát vracející ukazatel na bitmapu pro lepší přístup k prvkům fotografie.

Pokud byla fotografie pořízena úspěšně, dojde k její analýze. K procházení obrázku je použita bitmapová utilita obsažená ve třídě TBitmapUtil. Ta obsahuje speciální metody pro rychlý přístup k jednotlivým pixelům obrázku a je méně náročná než klasické procházení polem.

Z daného obrazu je vypočtena jeho průměrná světelnost a její výstupní hodnota je navrácena nadřazené třídě.

Touto nadřazenou třídou je třída CCameraController. Její funkcí je připravit kameru pro snímání ve chvílích kdy je potřeba, bez toho aby se nadřazený kontroler muset starat o detaily. Třída obsahuje metody na vytvoření instance kamery, její zrušení a zanalyzování výsledné hodnoty. S třídou CCameraEngine komunikuje pomocí callbacku definovaném v abstraktní třídě MPictureNotify. Po obdržení analyzovaných dat ukončí práci s kamerou a předá hodnotu kontroleru.

6.5.4 Práce se zvukem

O analýzu zvuku se stará modul reprezentovaný třídou CStreamRecorder, který dědí z třídy MMdaAudioInputStreamCallback kvůli přístupu k zvukových funkcím a callbacku oznamující aktuální stav zaznamenávání. Také je nutné naincludovat knihovny MMF a MDA obsluhující zvukové zařízení a audiostreamy.

Pokud je nutné zanalyzovat zvuk, vytvoří kontroler instancí třídy CStreamRecorder, která ve svém konstruktoru alokuje paměť pro nahrávání zvuku. Po zavolání metody Record se vytvoří instance třídy CMdaAudioInputStream, která dokáže nahrávat zvuk do paměťového bufferu a zavolá se její metoda Open. Pokud byl stream úspěšně zinicializován, je přes callback MaiscOpenComplete zaslána žádost o nastavení formátu a začne nahrávání. O průběhu nahrávání informuje callback MaiscBufferCopied. Po ukončení nahrávání je zavolána metoda na zanalyzování vstupního zvuku.

Pro analýzu zvuku je využit algoritmus FFT s prvočíselným rozkladem (definice viz. 4.7.2) vytvořený panem Laurentem de Soras[8] v licenci LGPL[9]. Tento algoritmus očekává na vstupu pole diskrétních vzorků zvuku a na výstupu vrací pole obsahující vypočítaný FFT. Z pole se vypočte energie zvuku pomocí vzorce 6.1, kde N je počet vzorků [58]. Poté se zašle kontroleru pro další zpracování.

$$E = \frac{1}{N} \sum_{n=0}^{N-1} |x(n)|^2 \tag{6.1}$$

[8] ke stažení na http://ldesoras.free.fr/src/FFTReal-1.03.zip
[9] Lesser General Public License

6.6 Implementace obsluhy GPS modulu

Přístup k GPS modulu je uskutečněn přes Location API. Location API obsahuje několik klíčových tříd potřebných pro správnou identifikaci polohy, Jedná se o RPositionServer, RPositioner a TPositionInfo.

V implementaci je vytvořena třída CLocation, která obsluhuje GPS modul díky výše zmíněným třídám. V konstruktoru jsou nejprve vytvořeny vlastnosti, podle kterých se má GPS modul chovat. Jsou zde definovány informace jako rychlost obnovování pozice, hodnota definující maximální dobu snažení o nalezení družit a další. Po nastavení je odeslána žádost o získání pozice. Ke zjištění pozice slouží metody GetLastKnownPosition, NotifyPositionUpdate třídy NotifyPositionUpdate. V případě zjištění souřadnice se informace předají přes callback PositionUpdatedL jinak se zavolá callback PositionLost informující o ztrátě signálu. Po obdržení souřadnice se informace předá GPS kontroleru definovaného třídou CLocationController.

Třída CLocationController se stará o správnou funkčnost GPS modulu a o správnou interpretaci získaných dat. Obsahuje metodu GPSsensorAviable na zjištění přítomnosti GPS modulu, které lze volat z jiných tříd. Dále metodu PositionUpdatedL, která přijme informace o změně polohy a získá aktuální informace o poloze obsažené ve třídě TPosition. Souřadnice zde zjištěné posílá do nadřazeného kontroleru pro další analýzu.

6.7 Kontroler

Kontroler je třída CController odpovídající za získání hodnot ze senzorů a modulů, za jejich analýzu a informování tříd starajících se o vyzvánění o výsledcích analýzy. Tato třída funguje jako mezivrstva mezi grafickou částí a částí obsluhující hardware mobilního telefonu.

Nejdůležitější metoda RunData se stará o zavolání správných senzorů podle parametrů zadaných v GUI. Pokud je potřeba zavolat senzor, nastaví příznak a zavolá metodu které jej zinicializuje a spustí. Metody odpovídající za spuštění senzorů jsou RunCamera, RunAccSenzor, RunSound a RunGPS. Dále je zde přítomna metoda na zastavení senzorů StopData a metoda na zanalyzování vzdálenosti z GPS modulu LocationResult. Tato metoda v případě začátku vyzvánění zjistí nejbližší souřadnici uloženou v programu, vypočte vzdálenost a tuto vzdálenost použije pro určení vyzvánění.

6.8 Nastavené vyzvánění

Třída starající se o obsloužení příchozího telefonátu se nazývá CRun. Obsahuje callback CallStatusChangedL ze třídy MCallCallBack informující o změně stavu telefonu v rámci hlasových služeb. Tento callback registruje jak příchozí, tak odchozí volání. Je tedy nutné reagovat na správné události. Události jsou definované ve třídě CTelephony.

Pokud je aktuální událost rovna EStatusRinging, pak telefon obdržel informaci o příchozím telefonátu a začne vyzvánět. Tomuto je nutné zamezit zavoláním metody, starající se o vypnutí veškerých projevů vyzvánění, ze třídy CMute. Následně je nutné zjistit telefonní číslo volajícího. K tomu slouží skupina metod ze třídy CTelephony, z nichž je nejdůležitější metoda GetCallInfo. Tato metoda vrátí informace o příchozím telefonátu, mezi nimi i aktuální telefonní číslo.

Jakmile se získá telefonní číslo, dojde ke spuštění senzorů a načtení profilu telefonu. Z profilu se zjistí typ vyzvánění a vibrací. Dále se načte uložený profil aplikace pro dané

číslo metodou GetRinginTone. Zjistí se informace o globálním nastavení aplikace a podle toho se upraví maximální a minimální hodnoty vyzvánění. Také se načte příslušné vyzvánění. Se spuštěním senzorů se načtou jejich parametry, které upravují citlivost senzorů, případně je vypínají v metodě LoadSenzorsSettings.

Po úspěšném načtení informací a získání potřebných dat ze senzorů, dojde k vyhodnocení. Vyhodnocení probíhá v metodě SetRinging. Výstupem jsou hodnoty hlasitosti vyzvánění, intenzity vibrací a stylu vyzvánění. O průběh vyzvánění se stará metoda StartRinging. Po ukončení vyzvánění se přehrávání vypne a systém čeká na další vyzvánění.

Trénování klasifikátoru probíhalo při vývoji programu. Hodnoty získané klasifikátorem obsaženém v metodě SetRinging se stanovily jako referenční a uživatel je může ovlivňovat jen pomocí změny citlivostí senzorů. Tento přístup byl zvolen z důvodu usnadnění ovládání programu uživatelem.

6.9 Vyzvánění a vibrace

Úprava vyzvánění se aplikuje pouze při aktivním profilu s názvem **Inteligentni** nebo **Intelligent**.

6.9.1 Přehrávání zvuku

Vyzvánění probíhá pomocí přehrávače zvuků implementovaného ve třídě CPlayerUtility. Tato třída využívá ke své práci třídu CMdaAudioPlayerUtility starající se v Symbian OS o přehrávání souborů různého formátu. Také dědí ze třídy MMdaAudioPlayerCallback kvůli informacím o aktuálním stavu přehrávání. Pro ovládání přehrávače jsou implementovány metody Play sloužící k zahájení přehrávání, Stop pro ukončení přehrávání, MapcInitComplete ve které se nastavují vlastnosti přehrávání, jako je hlasitost, opakování a zesilování zvuku. Po ukončení přehrávání je třída informována pomocí callbacku MapcPlayComplete.

6.9.2 Vibrace

Pro spouštění vibrací je vytvořena třída CVibraController dědící ze třídy MHWRMVibraObserver. Pro používání vibrací je nutné využít třídy CHWRMVibra ovládající vibrační mechanizmus v mobilním telefonu.

Pro aktivování vibrací je nutné zavolat metodu StartVibrationL ve které dojde k ověření hodnot a následně se metoda StartVibraL, která spustí vibrační mechanizmus. Po uplynutí maximální doby vibrací je třída CVibraController informována o ukončení vibrací, následně se vibrační mechanizmus spustí znova a pokračuje do nekonečna nebo ukončení vyzvánění.

6.10 Uložení dat

Protože si program potřebuje pamatovat různá nastavení, je nutné je nějakým způsobem ukládat. Udržovat si veškerý obsah dat v paměti není možné, minimálně z důvodu paměťových nároků. Také v případě vypnutí programu nebo mobilního telefonu by došlo ke ztrátám veškerých dat. Z tohoto důvodu se musel vymyslet permanentní způsob uložení dat. Jako nejlepší přístup je využít schopností Symbian OS. Symbian OS podporuje ukládání nastavení programů do souborů pomocí třídy CDictionaryFileStore. Tato třída dovoluje ukládat data pomocí streamů identifikovaných podle UID[10]. Každé UID je definováno v aplikaci

[10]Unique identifier = jednoznačně identifikující hodnota

a je jedinečné. Také jednotlivé záznamy jsou jedinečné a mají definovány své ID. Tento přístup má výhodu v jednoduchosti získávání dat. Pro získání záznamu stačí tedy zadat UID aplikace a ID záznamu. Záznam může obsahovat jakékoliv informace, tedy i vnořené třídy nebo také obrázky.

Pro ukládání a načítání dat byla vytvořena třída CSaDictionaryController. Jako nosič informací jednotlivých záložek byly vytvořeny následující třídy. Jedná se o CStCislaModel, CStGlobalDataModel, CStGPSDataModel a CStSenzorDataModel.

Třída CStCislaModel v sobě udržuje data obsahující informace o speciálním nastavení vyzvánění pro dané číslo. Pro každou položku v menu je vytvořen jeden záznam. Také je nutné vytvořit přetíženou metodu pro zápis a načtení dat ze streamů. Jedná se o ExternalizeL obsluhující zápis dat do streamu a o InternalizeL starající se o čtení ze streamů. Tvorbu těchto metod, jen s úpravami pro danou třídu, bylo nutné zopakovat u každé třídy.

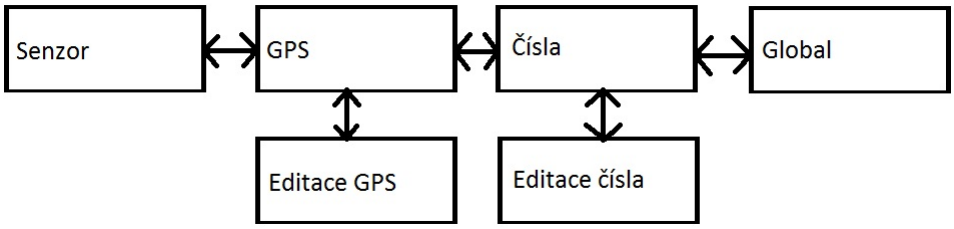
Pro uložení dat obstarávající globální nastavení je vytvořena třída CStGlobalDataModel. Třída CStGPSDataModel obstarává informace o souřadnicích GPS. Zbývá třída CStSenzorDataModel, která obsahuje nastavení senzorů.

Metody pro práci s uloženými daty jsou implementovány ve třídě CSaDictionaryController. Pro každou záložku v aplikaci jsou vytvořeny metody, které obstarávají zpracování dat před uložením (metody začínající předponou Save) a také po načtení (metody začínající předponou Load). Tyto dvě metody jsou ale postačující jen pro data, která se nepřidávají nebo neodebírají. Pro ostatní, které pracují se seznamy záznamů bylo nutné také naimplementovat metody starající se o vymazání již nepotřebných dat a metody zapisující aktuální počet záznamů na každý seznam. Mazání záznamů je důležité z hlediska udržování konzistence dat a zápis počtu záznamů zase pomáhá urychlit v orientaci záznamů. Odpadá tím počítání uložených položek a dochází tedy ke zkracování časové náročnosti.

6.11 Grafické uživatelské rozhraní

V aplikaci je nutné využívat uživatelských vstupů. K tomu slouží grafické uživatelské prostředí (dále i GUI). Implementace GUI probíhala za použití vestavěného editoru v Carbide.c++ nazvaného UI Designer. Celý editor lze ovládat myší. Jednotlivé obrazovky jsou tvořeny pouhým přetahováním elementů na reprezentaci okna. Jednotlivé prvky mají vlastní nastavovací vlastnosti. Jde například o rozsah povolených hodnot, implicitní nastavení nebo reakce na událost. Těmito reakcemi může být zmáčknutí klávesy, přepnutí obrazovky nebo uložení kontextu okna.

Návaznost obrazovek je znázorněna na diagramu návaznosti obrazovek (viz. 6.1). Jak lze vidět, program je složen ze čtyř základních obrazovek, mezi kterými lze přepínat pomocí kurzorových tlačítek. Jsou zde také umístěny dvě editační okna. Jedno pro úpravu a přidávání GPS souřadnic, druhé pro úpravu a přidávání vlastností telefonních čísel.



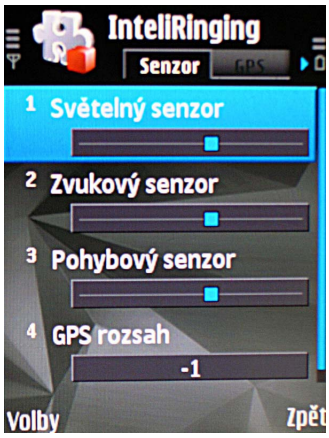
Obrázek 6.1: Diagram návaznosti obrazovek

6.12 Ovládání aplikace

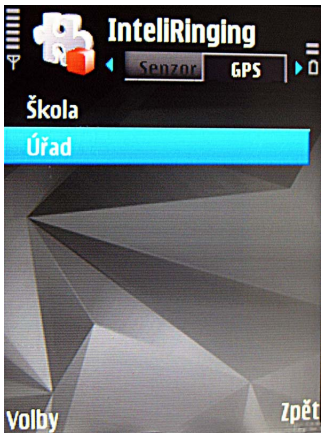
Aplikace se skládá z grafického uživatelského rozhraní a části ovládající senzory (viz. Diagram návaznosti obrazovek 6.1). Uživatel má přístup jen do části obsažené v GUI. Hlavní obrazovky mají shodné ovládací prvky na pravé tlačítko, které aplikaci pošle do pozadí. Také jeden ovládací prvek levého tlačítka je shodný se všemi hlavními obrazovkami. Jedná se o vypnutí aplikace s názvem *Exit*. Následující obrázky přibližují grafickou stránku aplikace.

První obrázek 6.2a ukazuje hlavní obrazovku s nastavením citlivosti senzorů. Při posunu posuvníku do levé části se citlivost zvyšuje, kdežto při posunu doprava se snižuje. Implicitní hodnota je tři a udává střední citlivost. V případě nastavení posuvníku na nulovou hodnotu se senzory neaktivují. Vypnutí senzorů ale nelze doporučit kvůli snížené schopnosti reagovat na okolní prostředí. Položka GPS má nastavený rozsah od -1 po 999. Jednotkou jsou metry a v případě nastavení hodnoty -1 je GPS čidlo odpojeno. Je nutné poznamenat, že při zapnutém GPS čidlu je v místech bez signálu, nebo s velice špatným signálem, doba před prvním zazvoněním velice dlouhá. V jistých případech může přesáhnout i osm sekund. Což odpovídá přibližně třem vyzváněcím tónům.

Druhý obrázek 6.2b reprezentuje seznam souřadnic GPS. Pokud jsou nějaké souřadnice přítomny, lze zmáčknutím tlačítka enter jednotlivé záznamy editovat. Také lze použít vlastnosti levého tlačítka pro editaci (*Edit*), přidání (*Přidat*) nebo smazání (*Smazat*) záznamu.



(a) Obrazovka s nastavením senzorů

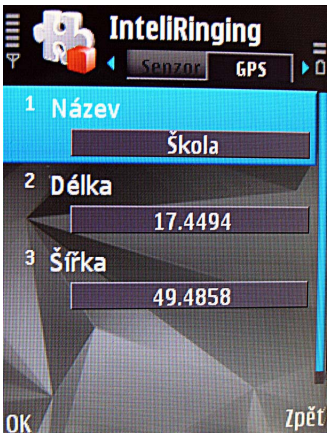


(b) Obrazovka se seznamem GPS souřadnic

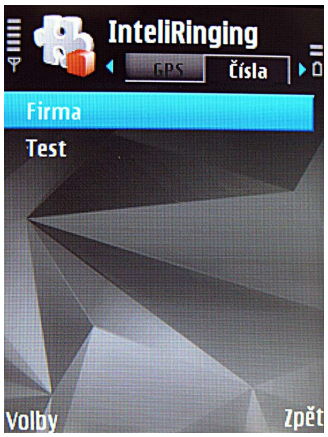
Obrázek 6.2: Obrazovky s nastavením senzorů a seznamem GPS souřadnic

Pokud uživatel zvolil editaci nebo přidání GPS záznamu, zobrazí se mu obrazovka s nastavením jednotlivých souřadnic a s určením názvu. Hodnoty jsou v případě přidání nové hodnoty předvyplněny vzorovými daty. Příklad uživatelských souřadnic je na obrázku 6.3a. Při přidávání souřadnic je nutné dávat pozor na zadání desetiné tečky místo čárky.

Třetí obrazovka v pořadí záložek s názvem *Čísla*, na obr.6.3b, slouží k zobrazení seznamu čísel obsahující upravené vlastnosti pro daná telefonní čísla. Vlastnosti ovládání jsou shodné s ovládáním obrazovky seznamu souřadnic GPS jen na místo úpravy GPS souřadnic se zde spouští editor telefonních čísel.



(a) Obrazovka na editaci GPS souřadnic



(b) Obrazovka se seznamem čísel

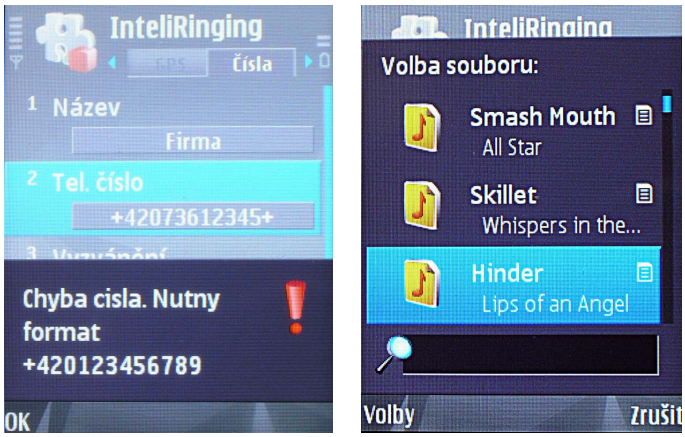
Obrázek 6.3: Obrazovky na editaci GPS souřadnic a seznamem čísel

V případě editování telefonních čísel se zobrazí obrazovka 6.5. První záznam je pojmenování položky, jak se zobrazí na přehledu čísel. Druhá je telefonní číslo, na které se bude vázat další úprava. Třetí položka reprezentuje zvukový soubor, který se bude při vyzvánění přehrávat. Následuje hlasitost vyzvánění. Na páté pozici je položka na úpravu intenzity vibrací, v případě, že je nastavena na nulu, jsou vibrace vypnuté. Poslední položka udává, jestli je nastavení prioritní. Pokud je *Vynucení vyzvánění* zaplé, nebere se ohled na nastavení profilu a vyzvánění se řídí pouze tímto nastavením.

(a) Část 1 (b) Část 2

Obrázek 6.4: Obrazovka na editaci čísel

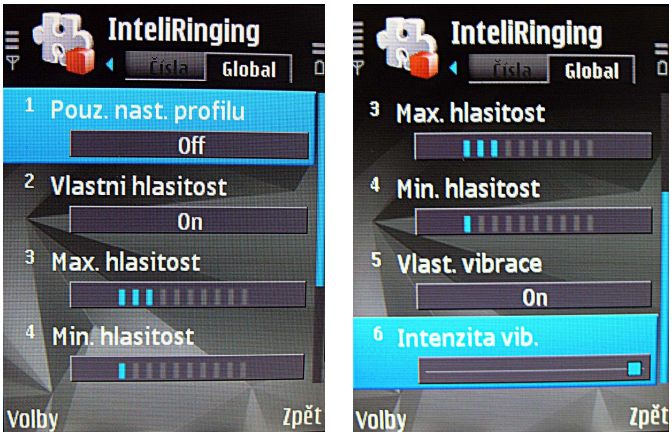
Jsou zde také implementovány algoritmy na kontrolu vstupních polí. Především jde o kontrolu správnosti telefonního čísla. Ukázky stylu zobrazení chybové hlášky a výběru vyzváněcího souboru jsou na následujících obrázcích 6.5.



(a) Hlášení chybného čísla (b) Výběr souboru

Obrázek 6.5: Obrazovka na editaci čísel

Poslední záložkou v aplikaci je globální nastavení (*Global*). Zde lze kompletně přenastavit maximální a minimální hodnoty vyzvánění a vibrací. Pro aktivaci tohoto nastavení je nutné zvolit *On* v položce *Pouz. nast. profilu*. Tato záložka byla přidána z důvodu nemožnosti používat zvukové a vibrační nastavení v případě, že je v profilu mobilního telefonu nastaveno vyzvánění na tiché nebo pípnutí. Tato záložka tedy dokáže simulovat toto profilové nastavení. Obrazovka je zobrazena na 6.6.



(a) Část 1 (b) Část 2

Obrázek 6.6: Obrazovka Globální nastavení

6.13 Analýza aplikace

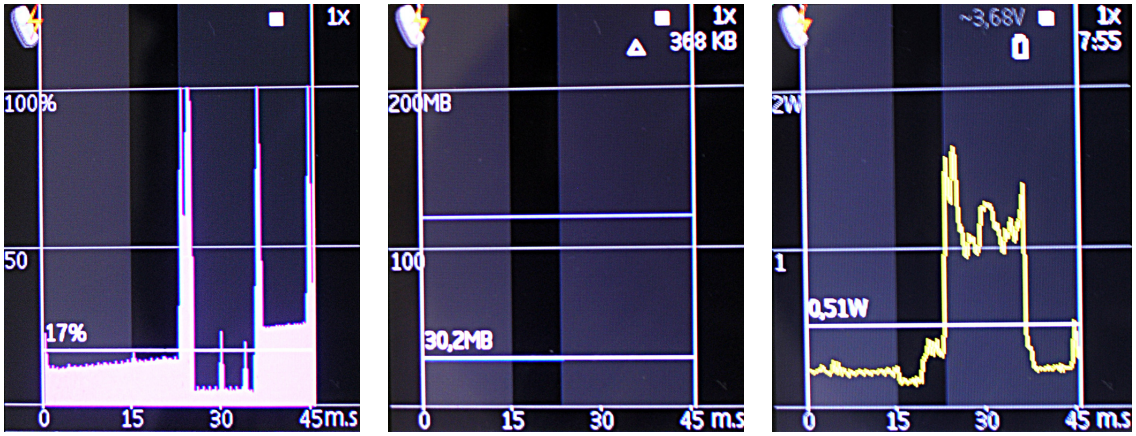
V této části je rozbor využití zdrojů vzhledem ke klidovému stavu mobilního telefonu. Jsou zde ukázány grafy spotřeby a výkonu při používání aplikace.

Testy probíhaly ve třech fázích, vždy se jednalo o příchozí volání. První test obsahoval jedno zavolání s vypnutým modulem GPS, druhý obsahoval dvě na sebe navazující zavolání, také s vypnutým GPS modulem. Třetí test probíhal se zapnutým GPS modulem a jednalo se pouze o jedno zavolání.

Grafy byly pořízeny pomocí aplikace *Energy Profiler* v. 1.22

6.13.1 Bez spuštěné aplikace

Pro porovnání s naměřenými výsledky se jako první zjistilo vytížení při vyzvánění bez zaplé aplikace. Grafy na 6.7 znázorňují toto vytížení.

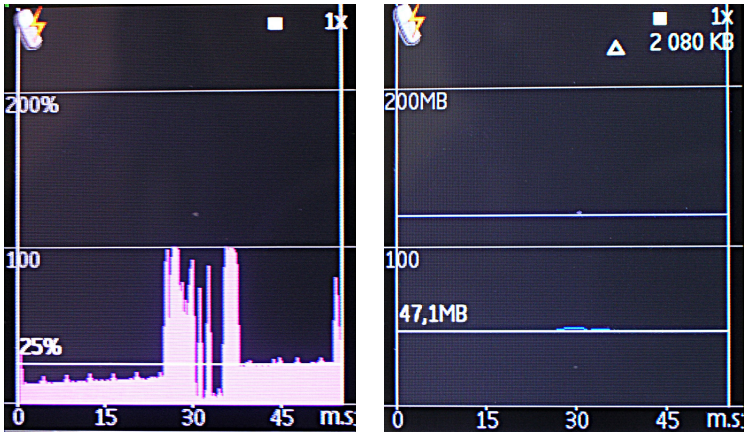


(a) Využití CPU (b) Využití RAM (c) Spotřeba W/ms

Obrázek 6.7: Souhrnné grafy - bez GPS

6.13.2 Test s jedním zavoláním a vypnutým GPS modulem

Graf na obrázku 6.8, *Využití CPU*, znázorňuje využití CPU. Jak je z grafu patrné, po detekování příchozího hovoru a spuštění všech senzorů a jejich analýz dojde k úplnému vytížení CPU v několika časových stavech. Nejprve dojde k zjištění a prvotní analýze akceleračního čidla a zvuku. Následně dojde k pořízení fotografie a analýze obrazu. Je to způsobenou delší inicializací fotografického senzoru. Po analýze je využití procesoru spotřebováno na přehrávání MP3 souboru.



(a) Využití CPU (b) Využití RAM

Obrázek 6.8: Využití CPU, RAM - bez GPS

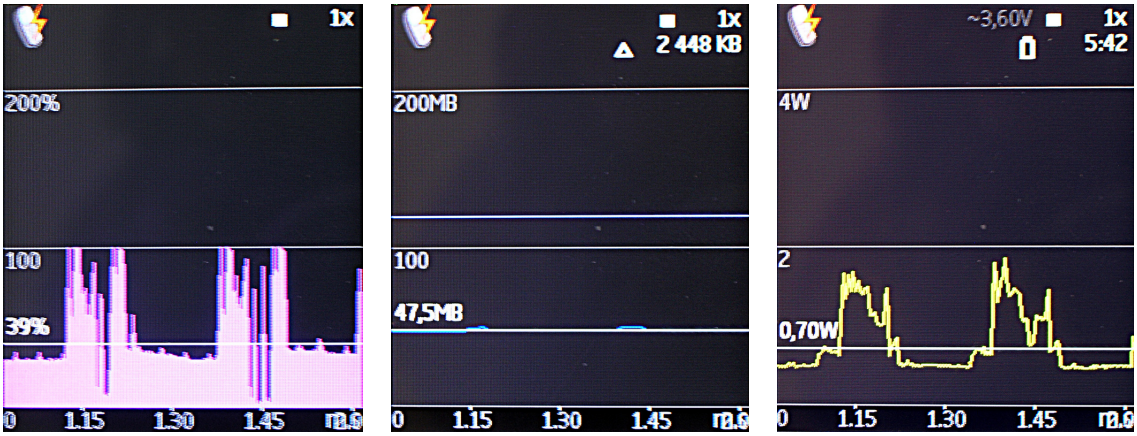
Druhý graf z obrázku 6.8, *Využití RAM*, znázorňuje využití paměti typu RAM. Jak je z grafu patrné, v případě zahájení volání dojde jen k minimálnímu zvětšení nároku na paměť RAM. Toto zvětšení je v řádech několika MB. Bohužel přesnější hodnoty se nepodařilo zjistit.

Následující graf na 6.9 zobrazuje změnu spotřeby energie v případě práce aplikace. Zde je již nárůst v grafu více zřetelný. Jak je vidět, nárůst spotřeby je více než dvojnásobný. Je ovšem nutné poznamenat, že k inicializaci spojení a komunikaci s buňkou GSM sítě je také nutné spotřebovat jisté množství energie (viz. 6.7). V porovnáním s grafy spotřeby bez aktivní aplikace se jedná o nárůst přibližně 0,25W.

Obrázek 6.9: Spotřeba W/ms

6.13.3 Test se dvěmi zavoláními a vypnutou GPS

Na grafech 6.10 je zobrazena výkonová analýza tzn. využití CPU, RAM a spotřeba energie po dvou vyzváněních. Na grafech jednotlivých vyzvánění jsou patrné minimální rozdíly.



(a) Využití CPU (b) Využití RAM (c) Spotřeba W/ms

Obrázek 6.10: Souhrnné grafy - bez GPS

6.13.4 Test s jedním zavoláním a zapnutým GPS modulem

Test slouží pro zjištění rozdílu ve spotřebě s prvním měřením. Výsledky ukázaly, že spotřeba při zapnuté GPS se dostala k hranici 3W, respektive 750mA. Také delší doba zpracování je z grafů patrná. Grafy jsou znázorněny na obrázcích 6.11, 6.11c.

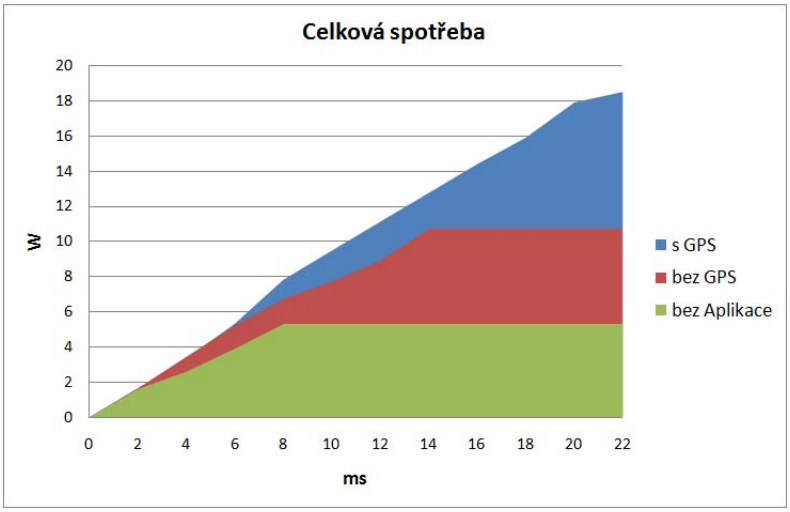
(a) Využití CPU (b) Využití RAM (c) Spotřeba W/ms

Obrázek 6.11: Využití CPU, RAM a spotřeba W/ms - s GPS

6.13.5 Vyhodnocení testů

Z naměřených hodnot vyplývá skutečnost, že po výkonnostní a paměťové stránce není aplikace příliš náročná. V případě vypnutého GPS modulu je i spotřeba energie v únosných hranicích. Problém začíná až při zaplém GPS modulu, kdy se spotřeba dostává nad 3W. Tento fakt snižuje výdrž baterie a nelze tedy doporučit využívání GPS modulu v místech, kde není možné dobít baterii mobilního telefonu.

Porovnání spotřeby energie je zobrazeno na grafu 6.12. V grafu je zobrazena spotřeba od chvíle detekování příchozího hovoru, až po zahájení vyzvánění. Celková spotřeba se zapnutým GPS modulem dosáhla téměř 19W, bez zapnutého GPS modulu 11W a bez použití programu 5W. Zvýšení spotřeby je úměrné počtu použitých senzorů a modulů telefonu.



Obrázek 6.12: Porovnání celkové spotřeby - bez spuštěné aplikace, bez GPS, s GPS

6.14 Známé problémy

V kapitole jsou zmíněny známé nestandardní chování aplikace a vysvětlení důvodů tohoto chování.

- **Vyzvánění a vibrace**
  První zásadní problém byl odhalen při aktivování vlastního vyzvánění a vibrací. V jistých případech není možné změnit nastavení profilu na nastavení aplikace. Jedná se o nemožnosti využívat vibrace pokud je mobilní telefon připojen k nabíječce, nebo přes datový kabel k počítači. Toto chování je vlastností Symbian OS a nelze obejít. Tato vlastnost je popsána v [38].

  Dále není možné programově vypnout vibrace pokud je zvolen druh vyzvánění tichý, případně jedno pípnutí. V nastavení tohoto profilu je také nemožné přehrávat vlastní zvuky, pokud je aktivní vyzvánění a dokonce také i zvuky nahrávat. Příčina této chyby není jistá. Existují dva možné důvody. První je nemožnost využití zdrojů kvůli nastavené prioritě, kdy telefon má nastavenou maximální prioritu pro zvukové zařízení. Druhý důvod je existence chyby v kódu Symbian OS. V případě použití tohoto vyzvánění nebude aplikací přehrán žádný zvuk, ani nebude použita analýza okolního hluku. Více informací lze získat na [56].

- **Akcelerační senzor**
  V akceleračním senzoru se při jistých pozicích mobilního telefonu občas vyskytuje chyba analýzy pohybu. Chyba nastává přibližně v 10% případů specifického umístění telefonu. Tuto chybu je možné odstranit snížením citlivosti čidla. Díky netypické pozici telefonu, kdy je mobilní telefon postaven na levou dolní hranu se zadním náklonem 45°, se ale snížení citlivosti nedoporučuje.

- **Světelný senzor**
  Ke světelnému senzoru lze přistoupit jen v SDK S60 3rd Edition FP2 a vyšších. Pokud mobilní telefon je postaven na starší platformě, je nutné využít integrovanou kameru. Problém je popsán v kapitole 3.4.2.

- **Emulátor**
  S využitím senzorů nelze používat emulátor mobilního telefonu na stolním počítači. Emulátor neobsahuje potřebné rozšíření pro senzory a nelze tedy testovat jejich chování. Ladění a testování je jedině možné na mobilním telefonu (tzn. on-device debugging).

- **On-device debugging**
  Ladění a testování na mobilním telefonu bylo omezeno typem a počtem využívaných oprávnění (capatibilities). V případě použití *Location capatibilities* nelze využít on-device debugging. K používání tohoto oprávnění je nutné aplikaci podepsat na portálu symbiansigned. Kvůli omezení používané bezplatné vývojářské licence nelze následně spustit on-device debugging. Tento samý problém nastane při využití více než pěti capatibilities. Problém lze vyřešit laděním jednotlivých modulů zvlášť.

- **Rozdíly v SDK**
  I přes velkou kompatibilitu SDK S60 3rd Edition, S60 3rd Edition FP1, S60 3rd Edition FP2 a S60 5th Edition není možné ke všem součástem telefonu přistupovat stejně. Nejzásadnější problémy jsou s přístupem k senzorům. SDK S60 3rd Edition

a S60 3rd Edition FP1 přistupují k senzorům pomocí Sensor API, kdežto S60 3rd Edition FP2 a S60 5th Edition pomocí Sensor Framework API. Tyto přístupy mezi sebou nejsou kompatibilní. Pro vytvoření aplikace využívající oba přístupy by bylo nutné implementovat pro každé SDK vlastní správu senzorů.

Kapitola 7

Závěr

Úkolem této práce bylo seznámení se s prostředky operačního systému Symbian. Speciálně s jeho verzí Symbian S60 a jeho implementací v současných smartphonech. Dalším bodem bylo zaměřit se na problematiku získávání znalostí s důrazem na jednoduchou klasifikaci proudů dat. S využitím výše zmíněných poznatků byla vytvořena aplikace, která využívá senzory k analýze okolního prostředí a dle nich upravuje styl vyzvánění.

Při plnění těchto požadavků bylo nutné nastudovat chování systému Symbian v různých situacích, využít veškeré vlastnosti Symbian S60 pro optimalizace a paralelizaci výpočtů a také překonat netypické chování Symbian S60 v některých situacích.

Z hlediska implementace bylo největším problémem získání dostatečných informací o práci senzorů, jelikož toto odvětví mobilních přístrojů není příliš využíváno. Také bylo nutné překonat nemožnost testování aplikace na emulátoru z důvodu neexistujících senzorových rozšíření pro emulátory. Taktéž politika přidělování certifikátů společnosti Symbian způsobovala značné nepříjemnosti při vývoji aplikace. Jednalo se hlavně o nemožnost ladění kódu za běhu, které bylo způsobeno omezením certifikátu pro nekomerční vývoj.

I přes veškeré problémy při vývoji splňuje výsledná aplikace veškeré požadavky zmíněné v zadání. Jediná odchylka od zadání je nevyužívání světelného senzoru kvůli nemožnosti přístupu k němu. Tento senzor byl nahrazen použitím fotografického modulu v mobilním telefonu. Přínosem této aplikace je využití veškerých vlastností mobilního telefonu a jeho chytré chování v různých situacích.

Možnosti dalšího vývoje aplikace jsou omezené. Je to zejména kvůli ukončení vývoje Symbian S60. Tento fakt byl v době zadání práce neznámý a objevil se až několik dnů před ukončením vývoje aplikace. Pro zprovoznění na nových telefonech s operačním systémem Symbian^3 bude nutné celou aplikaci přeprogramovat, jelikož Symbian S60 a Symbian^3 jsou z větší části nekompatibilní.

V případě pokračování vývoje aplikace pro Symbian S60 je možné rozšířit pole působení programu i na jiné funkce než je signalizace příchozích hovorů. Příkladem pro rozšíření může být inteligentní budík nebo podsvícení přizpůsobující se okolnímu prostředí.

Literatura

[1] Coulton, P.; Edwards, R.: *S60 Programming*. John Wiley & Sons Ltd, 2007, iSBN 978-0-470-02765-3.

[2] Han, J.; Kamber, M.: *Data Mining*. Morgan Kaufman, 2006, iSBN 978-1-55860-901-3.

[3] Harrison, R.: *Symbian OS*. Symbian Press, 2003, iSBN 978-0-470-85611-4.

[4] Harrison, R.: *Symbian OS*. Symbian Press, 2004, iSBN 978-0-470-87108-3.

[5] Harrison, R.: *Quick Recipes on Symbian OS*. Symbian Press, 2006, iSBN 978-0-470-99783-3.

[6] Kauppila, M.; Inkeroinen, T.; Pirttikangas, S.; aj.: MOBILE PHONE CONTROLLER BASED ON ACCELERATIVE GESTURING. 2008.

[7] Litchfield, S.: Assisted GPS and the future of smartphones [online]. http://www.allaboutsymbian.com/features/item/The_future_of_GPS-equipped_smartphones.php, 2007-06-27 [cit. 2010-05-21].

[8] Lukáš, R.: Klasifikace a predikce [online]. https://wis.fit.vutbr.cz/FIT/st/course-files-st.php/course/ZZN-IT/lectures/2008/05_klasifikace_predikce.ppt, 2008-11-03 [cit. 2010-05-13].

[9] MacKay, D.: *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2005.

[10] Nokia Corporation: Symbian C++ [online]. http://wiki.forum.nokia.com/index.php/Category:Symbian_C%2B%2B, 2006-01-19 [cit. 2010-05-11].

[11] Nokia Corporation: S60 2nd/3rd Edition: Differences in Features v1.5 [online]. 2006-10-16 [cit. 2010-05-11].

[12] Nokia Corporation: S60 Sensor Framework [online]. http://wiki.forum.nokia.com/index.php/S60_Sensor_Framework, 2006-11-18 [cit. 2010-05-11].

[13] Nokia Corporation: GPS API in S60 3rd Edition [online]. http://wiki.forum.nokia.com/index.php/GPS_API_in_S60_3rd_Edition, 2007-04-06 [cit. 2010-05-11].

[14] Nokia Corporation: Multimedia Framework Architecture in S60 Devices [online]. http://nds2.fds-forum.nokia.com/fdp/interface, 2007-05-08 [cit. 2010-05-11].

[15] Nokia Corporation: Audio Playback APIs [online]. http://wiki.forum.nokia.com/index.php/Audio_Playback_APIs, 2009-08-06 [cit. 2010-05-15].

[16] Nokia Corporation: GPS API in S60 3rd Edition [online]. http://wiki.forum.nokia.com/index.php/GPS_API_in_S60_3rd_Edition, 2009-08-25 [cit. 2010-05-15].

[17] Nokia Corporation: Media Recorder API [online]. http://wiki.forum.nokia.com/index.php/Media_Recorder_API, 2009-08-28 [cit. 2010-05-15].

[18] Nokia Corporation: Camera Application Engine API [online]. http://wiki.forum.nokia.com/index.php/Camera_Application_Engine_API, 2009-10-30 [cit. 2010-05-21].

[19] Nokia Corporation: Nokia Sensor APIs [online]. http://wiki.forum.nokia.com/index.php/Nokia_Sensor_APIs, 2009-12-23 [cit. 2010-05-15].

[20] Nokia Corporation: Location methods in S60 [online]. http://wiki.forum.nokia.com/index.php/Location_methods_in_S60, 2010-03-09 [cit. 2010-05-15].

[21] Nokia Corporation: Mobile J2ME Applications [online]. http://www.forum.nokia.com/Technology_Topics/Development_Platforms/Java.xhtml, [cit. 2010-05-21].

[22] Nokia Corporation: S60 Platform SDKs for Symbian OS, for C++ [online]. http://www.forum.nokia.com/info/sw.nokia.com/id/4a7149a5-95a5-4726-913a-3c6f21eb65a5/S60-SDK-0616-3.0-mr.html, [cit. 2010-05-21].

[23] Nokia Corporation: Symbian C++ [online]. http://www.forum.nokia.com/Technology_Topics/Development_Platforms/Symbian_C++/, [cit. 2010-05-21].

[24] Otrusina, L.: *Strojové učení v přirozeném jazyce [BP]*. FIT VUT v Brně, 2007.

[25] Pihan, R.: fotografovani.cz [online]. http://www.fotografovani.cz/art/fotech_df/rom_trouble1.html, [cit. 2010-05-12].

[26] Pokorný, J.: *Symbian S60 [BP]*. FIT VUT v Brně, 2009.

[27] Symbian: Symbian Signed [online]. https://www.symbiansigned.com/app/page, 2010-05-17 [cit. 2010-05-15].

[28] Symbian Foundation Limited: IM API Specification Document [online]. http://developer.symbian.org/main/documentation/reference/s3/pdk/specs/guides/IM_API_Specification/IM_API_Specification.html, 2006-11-07 [cit. 2010-05-20].

[29] Symbian Foundation Limited: MMS Client MTM API [online]. http://developer.symbian.org/main/documentation/reference/s3/pdk/specs/guides/MMS_Client_MTM_API_Specification/MMS_Client_MTM_API_Specification.html, 2006-11-07 [cit. 2010-05-20].

[30] Symbian Foundation Limited: Symbian3 [online]. http://developer.symbian.org/wiki/index.php/Symbian3, 2010-04-12 [cit. 2010-05-11].

[31] Symbian Foundation Limited: Qt & Symbian Platform Security [online]. http://developer.symbian.org/wiki/index.php/Qt_&_Symbian_Platform_Security, 2010-05-21 [cit. 2010-05-21].

[32] Symbian Foundation Limited: IP App protocols [online]. http://developer.symbian.org/main/source/packages/package/index.php?pk=160, [cit. 2010-05-20].

[33] Symbian Foundation Limited: IP Connection Management [online]. http://developer.symbian.org/main/source/packages/package/index.php?pk=129, [cit. 2010-05-20].

[34] Uko, T.: GPS navigace na FPGA [online]. https://dip.felk.cvut.cz/browse/pdfcache/ukot1_2008dipl.pdf, 2008 [cit. 2010-05-01].

[35] Univerzita Palackého Olomouc: (Diskrétní) Fourierova transformace [online]. http://apfyz.upol.cz/ucebnice/down/mini/fourtrans.pdf, 2003-12-11 [cit. 2010-05-12].

[36] Vojáček, A.: Jak pracují nové 3D MEMS akcelerometry [online]. http://hw.cz/Produkty/Nove-soucastky/ART1875-Jak-pracuji-nove-3D-MEMS-akcelerometry-Freescale-.html, [cit. 2010-05-21].

[37] WWW stránky: Analysis: What is a smart phone? [online]. http://networks.silicon.com/mobile/0,39024665,39156391,00.htm, 2006-02-12 [cit. 2010-05-11].

[38] WWW stránky: Wiki Nokia [online]. http://wiki.forum.nokia.com/index.php/KIS001304_-_Incoming_call_vibration_cannot_be_stopped_for_silent_ringing_type, 2009-03-10 [cit. 2010-05-21].

[39] WWW stránky: Python for S60 [online]. http://www.symbian-freak.com/downloads/freeware/cat_s60_3rd/descriptions/python/python_for_s60.htm, 2009-08-05 [cit. 2010-05-21].

[40] WWW stránky: Smartphone adoption as of February 2010 [online]. http://www.iptvevangelist.com/mobile/images/MobileOS_2_2010.jpg, 2010-02-01 [cit. 2010-05-21].

[41] WWW stránky: Akcelerometr [online].
http://wikipedia.infostar.cz/a/ac/accelerometer.html, 2010-03-25 [cit. 2010-05-15].

[42] WWW stránky: Symbian OS [online].
http://en.wikipedia.org/wiki/Symbian_OS, 2010-03-25 [cit. 2010-05-15].

[43] WWW stránky: Unified Modeling language [online].
http://cs.wikipedia.org/wiki/UML, 2010-03-25 [cit. 2010-05-15].

[44] WWW stránky: Fast Fourier transform [online].
http://en.wikipedia.org/wiki/Fast_Fourier_transform, 2010-04-19 [cit. 2010-05-11].

[45] WWW stránky: Fotodioda [online]. http://cs.wikipedia.org/wiki/Fotodioda, 2010-05-14 [cit. 2010-05-22].

[46] WWW stránky: Maemo OS [online]. http://en.wikipedia.org/wiki/Maemo, 2010-05-17 [cit. 2010-05-15].

[47] WWW stránky: iPhone OS [online]. http://en.wikipedia.org/wiki/IPhone_OS, 2010-05-20 [cit. 2010-05-21].

[48] WWW stránky: Openmoko [online]. http://en.wikipedia.org/wiki/Openmoko, 2010-05-20 [cit. 2010-05-21].

[49] WWW stránky: interval.cz [online].
http://interval.cz/clanky/j2me-v-kostce-prvni-midlet/, [cit. 2010-05-01].

[50] WWW stránky: Texas Instruments [online].
http://focus.ti.com/general/docs/wtbu/wtbugencontent.tsp?contentId=4612&navigationId=12034&templateId=6123, [cit. 2010-05-01].

[51] WWW stránky: Texas Instruments [online].
http://focus.ti.com/general/docs/wtbu/wtbuproductcontent.tsp?contentId=4663navigationId=12607&templateId=6123, [cit. 2010-05-01].

[52] WWW stránky: Tisch ITP [online].
http://itp.nyu.edu/physcomp/sensors/Reports/ADXL202Accelerometer, [cit. 2010-05-01].

[53] WWW stránky: Axis Communication [online].
http://www.axis.com/edu/axis/images/ccd.gif, [cit. 2010-05-11].

[54] WWW stránky: Make: [online].
http://blog.makezine.com/LightSensor-01-L.jpg, [cit. 2010-05-11].

[55] WWW stránky: Molecular Expressions [online]. http://www.microscopy.fsu.edu/primer/digitalimaging/images/cmos/cmoschipsfigure1.jpg, [cit. 2010-05-11].

[56] WWW stránky: Developer Discussion Boards [online].
http://discussion.forum.nokia.com/forum/showthread.php?t=186555, [cit. 2010-05-21].

[57] Zendulka, J.: Získávání znalostí z databází [online]. https://wis.fit.vutbr.cz/FIT/st/course-files-st.php/course/ZZN-IT/texts/ZZN.pdf, 2006-10-31 [cit. 2010-05-13].

[58] Černocký, J.: Parametrický popis řeči [online]. http://www.fit.vutbr.cz/~cernocky/oldspeech/labs/energie_nuly.pdf, [cit. 2010-05-24].

Seznam příloh

A: Uživatelská příručka - součástí textu
B: Datový nosič CD:

- soubor *README.TXT*
  - základní informace o práci, popis obsahu média
- adresář *application*
  - adresářová struktura projektu, vhodná pro import do Carbide.c++ IDE
  - v podadresáři *sis* je umístěna hotová nepodepsaná aplikace *InteliVyzvaneni.sis*
- adresář *doc*
  - automaticky vygenerovaná dokumentace zdrojového kódu pomocí nástroje *Doxygen*
- adresář *techzpr*
  - text této práce ve formě PDF
  - v podadresáři *src* jsou zdrojové soubory pro sytém LaTeX, obrázky a grafy

Dodatek A

Uživatelská příručka

Aplikace umožňuje přizpůsobování vyzvánění mobilního telefonu okolnímu prostředí za využití senzorů a modulů integrovaných v telefonu.

Uživatel má možnost nastavení citlivosti senzorů, změnit nastavení profilu telefonu, přiřadit GPS souřadnice pro ztlumení vyzvánění a upravit vlastnosti telefonního čísla.

Vyzvánění z programu je aplikováno v případě, že je v telefonu zvolen profil Intelligent nebo Inteligentni.

Ovládání aplikace v hlavních, needitačních, obrazovkách je následující:

- **Levé výběrové tlačítko**, s popisem Volby slouží k zobrazení menu upravující vlastnosti dané položky. Jedná se o:
  - Change - změní hodnotu aktuálně vybrané položky
  - Přidat - přidá záznam do aktuálního seznamu
  - Edit - upraví aktuálně zvolený záznam
  - Smazat - smaže aktuálně zvolený záznam
  - Konec - ukončí aplikaci
- **Pravé výběrové tlačítko**, s popisem Zpět slouží k přepnutí aplikace na pozadí.
- Klávesy **vlevo** a **vpravo** přepínají jednotlivé obrazovky. Aktuální obrazovka je zvýrazněna na panelu záložek.

Nastavení senzorů

- Po spuštění aplikace se zobrazí úvodní obrazovka obsahující nastavení senzorů. Na obrázku A.1a je zobrazeno výchozí nastavení senzorů.
- V případě nutnosti změny intenzity se při posunu posuvníku doleva zvyšuje citlivost senzoru, při posunu doprava se naopak citlivost snižuje.
- Při nastavení hodnoty na 0 je senzor vypnut.
- Nastavení GPS udává vzdálenost, při které se přestane používat zvukové vyzvánění. V případě nastavení vzdálenosti na -1 se GPS modul vypne.
- Vypínání jednotlivých senzorů, kromě GPS, se nedoporučuje z důvodu neobjektivního zanalyzování prostředí.

Globální nastavení profilu

- Záložka označená Global, zobrazená na obrázku A.1f, slouží k souhrnnému nastavení profilu. V případě aktivování tohoto profilu jeho první položkou, se ignoruje nastavení profilu vyzvánění v telefonu.
- Nastavení vyzvánění i profilu lze aktivovat nezávisle na sobě pomocí položky Vlastní hlasitost, respektive Vlastní vibrace.
- Pro hlasitost je možné nastavit maximální a minimální intenzitu.
- U vibrací lze nastavit pouze její maximální intenzitu.

Seznam GPS souřadnic

- Záložka označená GPS, zobrazená na obrázku A.1b, slouží k přehledné správě GPS souřadnic. Položky lze přidávat, editovat a odebírat pomocí levého tlačítka. Editace je také možná pomocí tlačítka enter.
- Pro přidávání a editaci se zobrazí adekvátní obrazovka.

Seznam čísel

- Seznam čísel s upravenými vlastnostmi je zobrazen na obrazovce s názvem Čísla (viz. obrázek A.1d). Pomocí této obrazovky lze čísla přidávat, editovat a odebírat pomocí levého tlačítka. Také je možné položky upravovat po stlačení tlačítka enter.
- Pro přidávání a editaci se zobrazí adekvátní obrazovka.

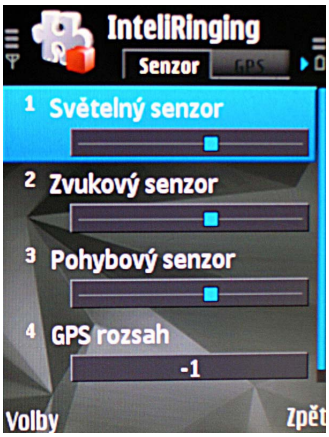
Editace GPS souřadnic

- Při požadavku na přidání, případně editaci GPS souřadnic je zobrazena obrazovka A.1c.
- Uživateli je umožněno upravit název položky a souřadnice délky a šířky.

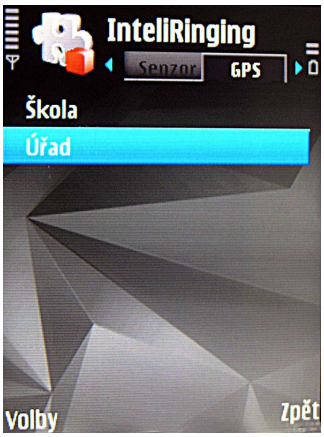
Editace čísel

- Pokud uživatel zvolil přidání čísla, případně jeho editaci je zobrazena obrazovka A.1e.
- Uživateli je umožněno upravit následující položky:
  - Název - název nastavení zobrazené v seznamu čísel
  - Tel. číslo - telefonní číslo, na které bude nastavení aplikováno
  - Vyzvánění - zvukový soubor, který bude přehráván při vyzvánění tohoto čísla. Zde je možné pomocí levého tlačítka nastavit vyzvánění z profilu telefonu po stisku volby Profilové. Také lze pomocí tohoto menu zobrazit jeho název s umístěním.
  - Hlasitost - udává maximální hlasitost vyzvánění
  - Vibrace - udává maximální intenzitu vibrací
  - Vynutit vyzvánění - pokud je aktivováno, aplikace se pokusí vynutit nastavení vyzvánění i vibrací na úkor nastavení profilu

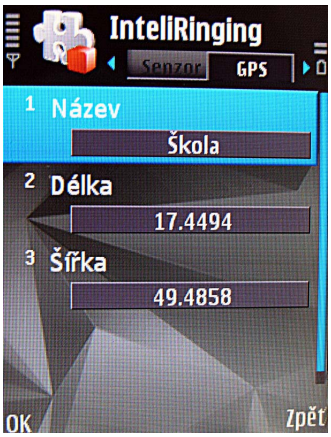
Pro aplikování upraveného vyzvánění musí být program aktivní. V případě příchozího hovoru dojde ke spuštění senzorů. Následně se informace ze senzorů vyhodnotí a spustí se vyzvánění, které může být upraveno podle příchozího telefonního čísla. Program neovlivňuje nastavení profilu v telefonu. Po vypnutí programu se vyzvánění řídí podle aktuálního vyzváněcího profilu.



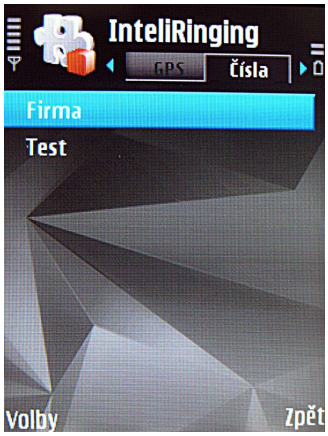
(a) Obrazovka s nastavením senzorů



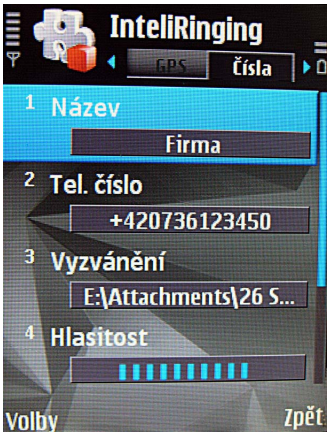
(b) Obrazovka se seznamem GPS souřadnic



(c) Obrazovka na editaci GPS souřadnic



(d) Obrazovka se seznamem čísel



(e) Obrazovka na editaci čísel

(f) Obrazovka Globální nastavení