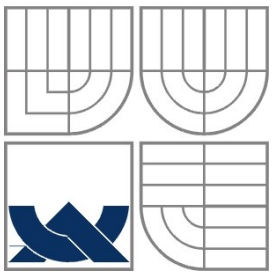
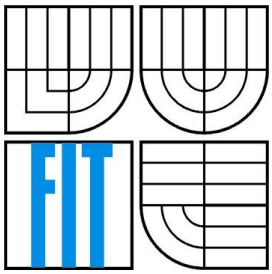


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

JEDNODUCHÝ IS PRO MOBILNÍ TELEFONY PRO EVIDENCI HOVORŮ

SIMPLE MOBILE PHONE IS FOR CALL EVIDENCE

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE
AUTHOR

MARTIN MATEJOVIČ

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. FRANTIŠEK ŠČUGLÍK

BRNO 2007

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav informačních systémů Akademický rok 2006/2007

Zadání bakalářské práce

Řešitel: **Matejovič Martin**
Obor: Informační technologie
Téma: **Jednoduchý IS pro mobilní telefony pro evidenci hovorů**
Kategorie: Počítačové sítě

Pokyny:

1. Seznamte se s možnostmi tvorby aplikací pro mobilní telefony
2. Zvolte jedno z vývojových prostředí s ohledem na rozšířenost mezi výrobci MT a toto prostředí si prostudujte
3. Navrhněte jednoduchý IS pro evidenci hovorů s možností zobrazovaní statistik a přehledů. Uvažujte možnost kalkulace cen hovorů na základě vyplněné tabulky cen hovorů na jednotlivé mobilní operátory
4. Navržený systém implementujte a otestujte jeho funkčnost

Literatura:

- Schiller, J.:Mobile Communications (2nd edition), Addison-Wesley 2003, ISBN 0-321-12381-6
- dle doporučení vedoucího

Při obhajobě semestrální části projektu je požadováno:

1. Seznamte se s možnostmi tvorby aplikací pro mobilní telefony
2. Zvolte jedno z vývojových prostředí s ohledem na rozšířenost mezi výrobci MT a toto prostředí si prostudujte

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním paměťovém médiu (disketa, CD-ROM), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Ščuglík František, Ing.**, UIFS FIT VUT
Datum zadání: 1. listopadu 2006
Datum odevzdání: 15. května 2007

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav informačních systémů
612 66 Brno, Božetěchova 2

L.S.

doc. Ing. Jaroslav Zendulka, CSc.
vedoucí ústavu

LICENČNÍ SMLOUVA
POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami

1. Pan

Jméno a příjmení: **Martin Matejovič**
Id studenta: 84246
Bytem: Školská 426, 956 05 Radošina
Narozen: 10. 03. 1986, Topoľčany
(dále jen "autor")

a

2. Vysoké učení technické v Brně

Fakulta informačních technologií
se sídlem Božetěchova 2/1, 612 66 Brno, IČO 00216305
jejímž jménem jedná na základě písemného pověření děkanem fakulty:

.....
(dále jen "nabyvatel")

Článek 1
Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):
bakalářská práce

Název VŠKP: Jednoduchý IS pro mobilní telefony pro evidenci hovorů
Vedoucí/školitel VŠKP: Ščuglík František, Ing.
Ústav: Ústav informačních systémů
Datum obhajoby VŠKP: .....

VŠKP odevzdal autor nabyvateli v:
tištěné formě počet exemplářů: 1
elektronické formě počet exemplářů: 2 (1 ve skladu dokumentů, 1 na CD)

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.
3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.
4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

Článek 2
Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizování výpisů, opisů a rozmnoženin.
2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.
3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti:
   - ☑ ihned po uzavření této smlouvy
   - ☐ 1 rok po uzavření této smlouvy
   - ☐ 3 roky po uzavření této smlouvy
   - ☐ 5 let po uzavření této smlouvy
   - ☐ 10 let po uzavření této smlouvy

   (z důvodu utajení v něm obsažených informací)
4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/ 1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

Článek 3
Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.
2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.
3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísni a za nápadně nevýhodných podmínek.
4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: .....

.....	Mrlepovic .....
Nabyvatel	Autor

Abstrakt

Táto bakalárska práca sa zaoberá programovaním aplikácií pre mobilné zariadenia, technológiou Bluetooth, implementáciou programov ktoré túto technológiu využivajú a technológiami a nástrojmi ktoré sa využívajú na mobilných zariadeniach.

Klúčové slová

mobilné aplikacie, Java 2 Micro Edition,Symbian, Bluetooth, CLDC, MIDP, NetBeans

Abstract

This bachelor thesis is about programming application for mobile devices, Bluetooth technology, implementation of applications which use it and generally about technologies and tools used for developing applications for mobile devices.

Keywords

mobile applications, Java 2 Micro Edition,Symbian, Bluetooth, CLDC, MIDP, NetBeans

Citácia

Martin Matejovič: Jednoduchý informačný systém pro mobilní telefón pro evidenci hovorů, **bakalárská práce, Brno, FIT VUT v Brne,** 2007

JEDNODUCHÝ IS PRO MOBILNÍ TELEFONY PRO EVIDENCI HOVORŮ

Prehlásenie

Prehlasujem, že túto bakalársku prácu som vypracoval samostatne pod vedením Ing. Františka Ščuglika.
Uviedol som všetky literárne pramene, z ktorých som čerpal.

.....
Martin Matejovič
15.mája 2007

Poďakovanie

Ďakujem vedúcemu bakalárskej práce Ing. Františkovi Ščuglíkovi za poskytnutú pomoc pri riešení problémov.

Obsah

1 Úvod

Mobilné telefóny sa stali bežnou súčasťou nášho každodenného života a ich vývoj dospel k tomu, že sú schopné používateľovi poskytnúť viac služieb ako iba telefonovanie.

Táto služba je, ale primárna a stále najviac využívaná. Používatelia mobilných telefonov, s predplatenými programami od operátorov často nemajú prehlad o tom, ako čerpajú svoje predplatené minúty. Taktiež spoločnosti, ktorých zamestnanci využívaju služobné telefóny, potrebujú mať prehľad o ich využívaní.

Preto je veľmi vhodne, mať k dispozícii nástroj ktorý dokáže jednoducho a rýchlo zhromažďovať informácie a vyhodnocovať používanie mobilného telefónu. O vytvorenie takého nástroja som sa pokúsil.

1.1 Vývoj mobilných aplikácií

Pri vývoji aplikácií na mobilný telefón sa vyskytujú ďalšie problémy oproti vývoju „klasických“ aplikácií pre PC.

Prvou prekážkou je testovanie a ladenie aplikácie. Na tento účel je treba využiť emulátor mobilného telefónu. Väčšina výrobcov telefónov poskytuje emulátor na svoje zariadenie. Tieto sa dajú stiahnuť na stránkach výrobcu. Ďalšou možnosťou je použitie univerzálneho emulátora od firmy SUN, dostupného zadarmo na stiahnutie zo stránok spoločnosti SUN.

Ďalším problémom je vyber vhodného programovacieho jazyka, ktorý by sme mali vybrať na základe zariadenia, pre ktoré je naša aplikácia určená.

1.1.1 Symbian/C++

1.1.1.1 Úvod

Programovanie aplikácií v jazyku C++ je možné v prostredí Symbian. Operačný systém(OS) Symbian patrí momentálne k najrozšírenejším operačným systémom na poli mobilných zariadení. Tento systém využíva vo svojich telefónoch najmä spoločnosť Nokia, ale aj iný popredný výrobcovia. Symbian je proprietárny operačný systém, vytvorený pre mobilné zariadenia, ktorý funguje výlučne na architektúre ARM. Symbian, ako bežné desktopové operačne systémy podporuje multitasking, delenie behu aplikácie do vlákien (tzv. multithreading) a taktiež ochranu pamäte. Je však prispôsobený na fungovanie na zariadeniach s obmedzenými zdrojmi, ktoré môžu bežať v kuse mesiace alebo roky.

Veľký dôraz je kladený na správu pamäte a taktiež na správu súborov. Programovanie pre tento OS je založené na zachytávaní udalostí.

1.1.1.2 Štruktúra

Najnižšia vrstva Symbianu sa skladá z jadra a užívateľských knižníc, ktoré umožnujú užívateľským aplikáciam prístup k jadru. Symbian je založený na architektúre mikrojadra, čo znamená, že v jadre sa nachádzajú iba tie najnutnejšie veci. Obsahuje plánovač a správcu pamäte, ale žiadnu správu siete alebo súborov. Správca súborov využíva DOSosvský pohľad na súborový systém (každá jednotka ma pridelené písmeno na identifikáciu a na oddeľovanie adresárov sa používa lomítko) Správca súborov v Symbiane podporuje viacero systémov súborov vrátane stále rozšíreného systému FAT32.

Na ďalších vrstvách sú systémové knižnice a veľký sieťový a komunikačný systém, ktorý zabezpečuje všetko čo sa týka komunikácie zariadenia. Tento systém taktiež zastrešuje Bluetooth, IrDA a USB.

1.1.1.3 Vývoj na Symbiane

Existuje veľa platforiem založených na operačnom systéme Symbian, medzi najvýznamnejšie patria UIQ (napr. Sony Ericsson P990) alebo S60 (Nokia N70).

Pri vývoji aplikácie pre telefón Nokia je vhodné využiť nástroj Carbide.c++, ktorý je voľne dostupný na stránkach Symbianu alebo Nokia. Tento nástroj sa dá jednoducho pri inštalácií integrovať do vývojového prostredia, napríklad Eclipse alebo NetBeans a poskytuje rôzne funkcie ako jednoduchý nástroj na tvorbu užívateľského rozhrania, nástroj na vytváranie toku aplikácie alebo ladenie pri páde aplikácie.

1.1.2 Java 2 Micro Edition

Java 2 Micro Edition (J2ME) je verzia populárneho programovacieho jazyka Java od spoločnosti Sun Microsystems určená pre trh s mobilnými zariadeniami ako sú mobilné telefóny, pagery alebo PDA. Tuto technológiu od jej vzniku podporuje množstvo spoločností ako Palm, Nokia, Motorola a iné.

J2ME poskytuje riešenie sieťových aplikácií pre malé zariadenia a tiež výrobcom poskytuje možnosť vyvíjať nové aplikácie pre svojich zákazníkov. Veľkou výhodou J2ME je jej kompatibilita s rôznymi prostrediami.

V princípe pohľadu J2ME poskytuje nasledovné komponenty:

- séria javovských virtuálnych strojov, každý sa uplatňuje na inom type zariadenia
- skupina knižníc a API, ktoré sú spustiteľné na každom virtuálnom stroji
- nástroje pre vývoj a nastavenie zariadení

Prvé dve komponenty tvoria *pracovné prostredie Javy*. Jeho centrum tvorí javovský virtuálny stroj, ktorý pracuje na hostiteľskom operačnom systéme. Nad nim je špecifická konfigurácia J2ME zložená z knižníc. Tie zaisťujú základné funkcie vychádzajúce zo zdrojov daného zariadenia. Vrchol konfigurácie tvoria jeden či viac profilov. Prehľadne je to možné vidieť na nasledovnom obrázku.

Pre vývoj vlastných aplikácií sú pre nás najpodstatnejšie vrchné dve vrstvy.

1.1.2.1 Konfigurácia

Mobilné telefóny, organizéry a ďalšie zariadenia sa líšia funkciami a vlastnosťami. Často ale používajú podobné procesory a majú aj podobné množstvo pamäte. Preto boli vytvorené *konfigurácie*. Konfigurácie definujú členenie produktov založené na množstve pamäti a výkonu procesoru. Akonáhle sú tieto informácie k dispozícii, konfigurácia zistí nasledujúce údaje:

- podporované rysy jazyka Java
- podporované rysy virtuálneho stroja Javy
- podporované knižnice a API

V súčasnej dobe existujú pre J2ME dve štandardné konfigurácie: CLDC (Connected Limited Device Configuration) a CDC (Connected Device Configuration)

1.1.2.1.1. CDC

Konfigurácia CDC je určená pre výkonné zariadenia, ktoré su občas pripojené do siete. Patria sem prídavné zariadenia, domáce spotrebiče alebo navigačné systémy pre vozidlá. CDC obsahuje plnú podporu virtuálneho stroja Javy, podobnému tomu, ktorý sa dnes používa v J2SE. Rozdiel je iba v dostupnom množstve pamäte a zobrazovacích schopnostiach.

Požiadavky na zdroje v CDC:

- zariadenie je vybavené 32-bitovým procesorom
- zariadenie má 2 MB a viac pamäte pre Javu. Zahrňuje sa RAM i pamäť flash alebo ROM
- zariadenie vyžaduje plne funkčný virtuálny stroj Java 2 „Blue Book“
- zariadenie je pripojené k niektorému typu siete, často bezdrôtovým pripojením
- zariadanenie môže mať aj prepracované užívateľské rozhranie, to však nie je nevyhnutné

1.1.2.1.2.CLDC

Druhý typ konfigurácie je pre J2ME častejší. Táto konfigurácia udáva oveľa menšie požiadavky na zariadenie ako CDC. Tieto požiadavky sú:

- zariadenie môže mať celkom 160 až 512 kB pamäte pre prostredie Java, vrátane RAM
- zariadenie môže mať obmedzený zdroj energie, spravidla batéria
- zariadenie je prepojiteľné s niektorým typom siete, často bezdrôtovým spojením
- zariadenie môže mať pomerne prepracované užívateľské rozhranie

Rozdelenie zariadení podľa konfigurácie zobrazuje aj nasledovný obrázok:

1.1.2.2 Profily

J2ME umožňuje definovať javovské prostredia pre rôzne produkty tím, že zavádza *profily*. Profil je sada programových rozhraní (API) tvoriacich nadstavbu konfigurácie. Profil poskytuje aplikácií prístup k vlastnostiam, ktoré sú špecifické pre dané zariadenie, na ktorom beží. V súčastnosti sú k dispozícií napríklad tieto profily:

- **MIDP**

  MIDP je navrhnutý pre prácu s CLDC a poskytuje sadu API použiteľných v mobilných zariadeniach, ako sú mobilné telefóny alebo pagery. MIDP obsahuje triedy pre tvorbu užívateľských rozhraní, trvalé ukladanie dát a prácu zo sieťou. Ďalej obsahuje štandardizované pracovné prostredie, ktoré umožňuje nami vytvorené aplikácie nahrávať do koncového zariadenia. Malé aplikácie, ktoré bežia pod MIDP, nazývame *Midlety.*

- **PDA**

  Profil PDA je založený na CLDC a poskytuje API pre užívateľské rozhrania a API pre ukladanie dát v príručných zariadeniach.

- **Základný profil**

  Základný profil rozširuje programové rozhranie, ktoré poskytuje CDC, ale neobsahuje žiadne API pre tvorbu užívateľského rozhrania. Ako už názov naznačuje tento profil slúži ako základ pre ďalšie profily.

- **Osobný profil**

  Osobný profil rozširuje možnosti Základného profilu o grafické užívateľské rozhranie, na ktorom sa dajú spustiť applety pre Java Web.

1.1.2.3 Virtuálny stroj Javy

Pri každej z konfigurácií je potrebný iný javovský virtuálny stroj. Virtuálny stroj pre CLDC je oveľa menši ako virtuálny stroj pre CDC. Virtuálny stroj pre CLDC sa vola KVM (Kilo Virtual Machine).

- **KVM**

  KVM je kompletné javovské prostredie pre malé zariadenia. Jedná sa o kompletný javovský stroj podla špecifikácie, až na malé odchýlky a prispôsobenia, ktoré sú nutné pre fungovanie na malých zariadeniach. Je od samého základu vytvorený pre špecialne pre malé zariadenia s obmedzenými zdrojmi a malou pamäťou. KVM bol pôvodne vytvorený v laboratóriach Sun Microsystems ako výskumný projekt pod názvom „Spotless“.Cieľom bolo vytvoriť virtuálny stroj Javy pre Palm.

- **CVM**

  CVM je vytvorený pre väčšie zariadenia, napríklad pre tie, ktoré využívajú CDC. Podporuje všetky rysy virtuálneho stroja Java 2 verzie 1.3 v oblasti zabezpečenia a

RMI (Remote Method Invocation) Implementácia odkazov, ktoré sú v súčastnosti k dispozícií od Sun Microsystems, funguje na Linuxe a VxWorks. Su dostupne na stránke J2ME.

1.2 Bluetooth

Elektronické zariadenia môžu byť prepojené rôznymi spôsobmi. Medzi najčastejšie spôsoby patri klasické káblové spojenie, infračervený prenos alebo spojenie techonológiou Bluetooth. Výhodou Bluetooth spojenia je, že nie sme obmedzovaný typom konektoru ako pri káblovom spojení a taktiež máme dostupnú väčšiu mobilitu ako pri infračervenom spojení vďaka väčšiemu dosahu.

Bluetooth funguje na štandardnej frekvencií 2,4 GHz, čo zaisťuje kompatibilitu signálu medzi rôznymi Bluetooth podporujúcimi zariadeniami.

1.1.1.1 História

„Bluetooth“ bola prezývka Haralda Blatlanda, Dánskeho kráľa, ktorý vládol v rokoch 940 až 981 a zjednotil Dánsko a časť Nórska pod svoju nadvládu.

Bezdrôtová technlógia Bluetooth bola pôvodne navrhnutá v roku 1994 švédskou spoločnosťou Ericsson, ako druh komunikácie mobilných zariadení na krátku vzdialenosť. V roku 1998 spoločnosti Ericsson, IBM, Intel, Nokia a Toshiba vytvorili konzorcium Bluetooth Special Interest Group (BSIG) za účelom vývoja voľne dostupnej technológie pre komunikáciu na krátku vzdialenosť. Odvtedy sa viac ako 2000 spoločností pripojilo k BSIG, vrátane výrobcov mobilných telefónov, PDA a osobných počítačov.

1.1.1.2 Bluetooth vs. IrDa

Domáce elektronické spotrebiče ako televízor alebo DVD prehrávač využívajú paprsok infračerveného spektra. Infračervene spektrum je spoľahlivé a jednoducho zabudovateľné do zariadenia, ale má aj svoje negatívne stránky:

- odosielateľ signálu musí mať priamy výhľad na príjemcu
- odosielateľ môže na nemôže posielať naraz na viacero zariadení

Z týchto nevýhod, ale plynú výhody:

- interferencia je nezvyklá
- doručenie správy je spoľahlivé, aj keď je v miestnosti viac zariadení, ktoré podporujú infračervený prenos, nakoľko odosielateľ ju dokáže naraz odoslať iba jednému.

1.1.1.3 Bluetooth vs. IEEE 802.11b

Napriek tomu, že ako Bluetooth tak aj IEEE 802.11b sú bezdrôtové komunikačné protokoly pracujúce na frekvencií 2,4GHz, je dôležite nevidieť Bluetooth ako náhradu IEEE 802.11b. Oba sú vytvorené na iné účely.

IEEE 802.11b je protokol vytvorený na spájanie relatívne veľkých zariadení s veľkou rýchlosťou a väčšou napájacou energiou, ako sú napríklad laptopy alebo bežné stolové počítače. Zariadenia dokážu komunikovať rýchlosťou až 54 Mbit/sek, na vzdialenosť až 100 metrov.

Naopak Bluetooth je vytvorený na komunikáciu malých zariadení ako mobilné telefóny alebo PDA komunikujúcich nižšou rýchlosťou (1 Mbit/sek) a na kratšiu vzdialenosť (10 metrov), čo znižuje nároky na spotrebu energie.

Ďalším podstatným rozdielom je, že IEEE 802.11b nebol vytvorený na prenos hlasovej komunikácie, zatiaľ čo Bluetooth podporuje oba prenosy, či už hlasový alebo dátový.

1.1.1.4 Vlastnosti Bluetooth

Hlavnými vlastnosťami Bluetooth sú:

- Bluetooth je bezdrôtový a automatický. Nepotrebuje na prevádzku žiadne káble ani nič špecialneho na vytvorenie spojenia. Zariadenie nájde partnera na komunikáciu automaticky a začne komunikovať bez zásahu užívateľa (samozrejme okrem prípadu kedy je potrebná autentifikácia)
- Bluetooh komunikácia mení frekvenciu, čo zaručuje že komunikácia je bezpečnejšia a ťažšie sa zachytáva potencionálnym útočníkom.
- Signál dokáže prejsť cez stenu, čiže odosielateľ nemusí byť v priamom dohľade príjmateľa.

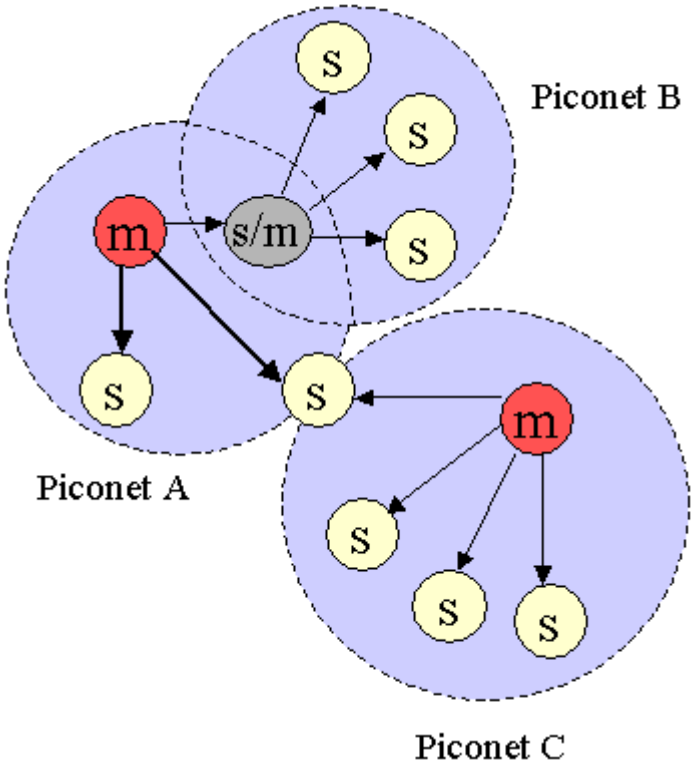
1.1.1.5 Aplikácie využívajúce Bluetooth

Technológia Bluetooth môže byť využitá napríklad pri týchto typoch aplikácií:

- Prenos súborov
- Synchronizácia zariadení
- Výbava na telefonovanie do auta
- Ovládanie spotrebičov v domácnosti
- Ovládanie osobného počítača pomocou mobilného telefónu

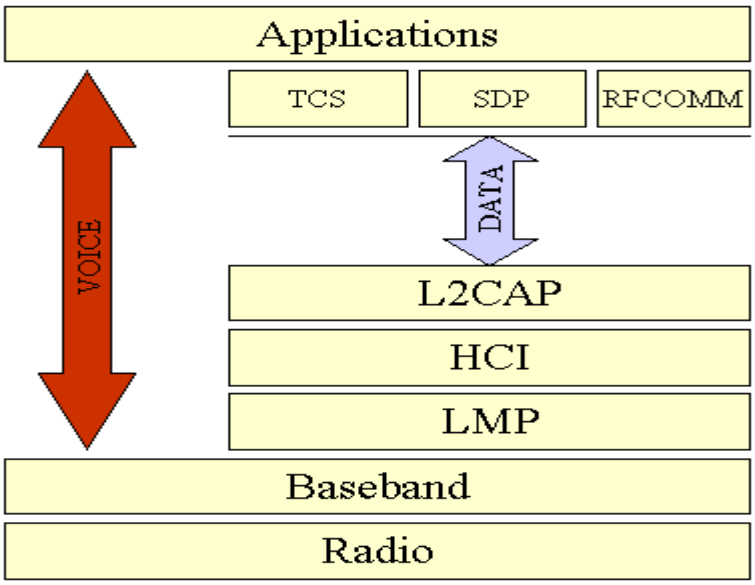
1.1.1.6 Topológia siete Bluetooth

Bluetooth zariadenia sú organizované do skupín nazývaných *piconety*. Piconet pozostáva z jedného zariadenia typu *master* a až siedmych zariadení typu *slave*. Master a jedno zariadenie slave využíva priame spojenie pre komunikáciu (tzv. point-to-point), pri komunikácií s viacerými zariadeniami sa využíva spojenie point-to-multipoint. Zariadenie master inicializuje spojenie. Zariadenie v jednom piconete, dokáže tiež komunikovať so zariadením v inom piconete. Topológia siete je zobrazená na nasledujúcom obrázku:



1.1.1.7 Arhchitektúra Bluetooth

Popis vrstiev znázornených na obrázku:



- *Radio*: zabezpečuje fyzickú vrstvu komunikácie. Za účelom predídenia interferencie s ostatnými zariadeniami sa využíva *frequency hopping,* na 79 kanáloch (od 2,402 do 2,480 GHz). Kanály sa menia 1600 krát za sekundu. Štandardný dosah je od 10 cm do 10 m, zvyšovaním napätia sa dá rozšíriť až na 100 m.
- *Baseband:* tato vrstva je zodpovedná za kontrolu a posielanie dátových paketov cez radiovu linku. Táto vrstva zaisťuje Synchrónnu linku (SCO) využívanú na prenos hlasu a Asynchrónnu linku (ACL) pre prenos dát.
- *Link Manager Protocol (LMP):* používa linky zostavené na Baseband, na zostavenie spojenia a udržiavanie piconetov. LMP má taktiež zodpovednosť za autentifikáciu, bezpečnostné služby a sledovanie kvality služieb.
- *Host Controler Interface (HCI):* je pomyselnou čiarou deliacou software a hardware. L2CAP a ostatné vrstvy nad ňou sú softwarovo implementované, LMP a nižšie vrstvy hardwareovo. HCI je ovládač, ktorý tieto dve vrstvy spája a umožňuje im spolu komunikovať. V niektorých prípadoch HCI nie je vyžadované.
- *The Logical Link Control and Adaptation Protocol (L2CAP)* : získava dáta z aplikácie a adaptuje ich na formát vhodný pre Bluetooth.

1.1.1.8 Vytvorenie Bluetooth spojenia

Ak zariadenie nie je pripojené do piconetu, je v pohotovostnom režime. V tomto režime zariadenie vyčkáva na správu každých 1,28 sekundy. Ak si niektoré zariadenie želá ustanoviť spojenie s iným, vyšle 16 identických správ na 16 frekvenciach. Ak *slave* neodpovedá, *master* zmení frekvencie a správu pošle na iných 16. Ak *master* nepozná adresu *slavea*, správu musí predchádzať tzv. pátracia sprava (*inquiry message*) . Keď *slave* odpovie na správu, *master* môže zahájiť prenos dát alebo hlasu.

Tu je príklad vytvorenia spojenia:

1. V novom prostredí zariadenie automaticky inicializuje vyhľadávanie prístupových bodov. Všetky dostupné prístupové body odpovedajú svojou adresou a zariadenie si niektorý z nich vyberie.
2. *Page* proces zosynchronizuje zariadenie s prístupovým bodom.
3. Vytvorenie linky. LMP vytvorí linku s prístupovým bodom.
4. Vyhľadávanie služieb. LMP používa *Service Discovery Protocol (SDP)*, na vyhľadanie služieb dostupných na prístupovom bode.
5. Vytvorenie *L2CAP* kanálu. LMP využije informácie získané pomocou *SDP* na vytvorenie *L2CAP* kanálu k prístupovému bodu. Aplikácia môže tento kanál využívať priamo, alebo využiť *Radio Frequency Communications Protocol (RFCOMM),* ktorý môže bežať nad *L2CAP. RFCOMM* emuluje sériovú linku.
6. Vytvorenie *RFCOMM* kanálu. V závislosti na potrebách aplikácie, je vytvorený *RFCOMM* kanál. Vytvorenie *RFCOMM* umožní existujúcej aplikácií využívajúcej sériový port pracovať cez Bluetooth.
7. Overenie: Toto je jediný krok, ktorý vyžaduje zásah užívateľa. Ak prístupový bod vyžaduje prihásenie, vyšle sa požiadavka na autentifikáciu a užívateľ bude vyzvaný na zadanie PIN pre prístup k službe.
8. Prenos dát

1.1.1.9 Bluetooth profily

Bluetooth profily sa využívajú na komunikáciu medzi rozdielnymi Bluetooth zariadeniami a tiež na komunikáciu aplikácií od rôznych výrobcov využívajúcich Bluetooth. Všetky Bluetooth profily sú popísané v Bluetooth špecifikácií, tu sú niektoré z nich vybrané:

- *Generic Acess Profile* definuje proces spájania a vyhľadávania zariadení a spravovanie linky. Tiež definuje procesy spojené s rozdielnymi bezpečnostnými modelmi. Minimálne tento profil musia podporovať všetky Bluetooth zariadenia
- *Service Discovery Application and Profile* definuje vlastnosti a procesy aplikácie v Bluetooth zariadení na vyhľadávanie registrovaných služieb a získavanie informácií o nich.
- *Serial Port Profile* definuje požiadavky na Bluetooth zariadenie, ktoré potrebuje emulovať pripojenie cez sériový port s využitím protokolu *RFCOMM.*
- *LAN Acess Profile* definuje ako môže Bluetooth zariadenie využívať služieb LAN pomocou *PPP*
- *Synchronization Profile* definuje požiadavky na aplikáciu, ktorá chce synchronizovať pomocou Bluetooth dáta na dvoch alebo viacerých zariadeniach

1.1.1.10 Zabezpečenie Bluetooth

Bezpečnosť je realizovaná troma spôsobmi:

- pseudo-náhodné menenie frekvencie
- autentifikácia
- kryptovanie

Všetky Bluetooth zariadenia musia mať implementovaný *Generic Acess Profile.* Tento profil definuje bezpečnostný model, ktorý zahrňuje tri režimy:

1) Režim 1: nezabezpečený režim komunikácie. Nie je využívaný žiadny spôsob zabezpečenia
2) Režim 2: ak zariadenie funguje v tomto režime, žiadny spôsob zabezpečenia sa nepoužíva kým nie je vytvorený komunikačný kanál
3) Režim 3: pri tomto režime sú bezpečnostné procedúry inicializované ešte pred vlastným linkovým spojením

2 Návrh

2.1 Vývojové prostredie NetBeans

.Vývojové prostredie treba zvoliť v súlade s jeho komplexnosťou a možnosťami, aké nám ponúka. Vzhľadom na to, že ide o aplikáciu pre mobilný telefón, je pri vývoji potrebných viac nástrojov ako pri vývoji aplikácie pre bežný osobný počítač. Na moju prácu som si zvolil vývojové prostredie NetBeans.

NetBeans je multiplatformové vývojové prostredie určené najmä pre vývoj v jazyku Java, ale pomocou zásuvných modulov sa dá rozšíriť aj na iné jazyky.

História

NetBeans sa začal vyvíjať v roku 1997 ako školský projekt na Karlovej univerzite v Prahe. Neskôr sa okolo projektu sformovala spoločnosť, ktorá vyvíjala komerčne vývojové prostredie NetBeans IDE, ktorú v roku 1999 kúpil Sun Microsystems. Sun o rok neskôr uvoľnil NetBeans ako *open-source.*

Použitie

Prvou z vecí ktorú musí naše vývojové prostredie obsahovať je generátor balíčkov, vo forme ktorých budeme aplikáciu do telefónu nahrávať. Tu nám často pomôžu zásuvné moduly, ktoré existujú k integrovaným vývojovým prostrediam.

Ďalším potrebným vybavením je aplikácia alebo nástroj, ktorým našu preloženú aplikáciu nahráme do telefónu. V tomto nám často pomôže software originálne dodaný od výrobcu telefónu, ktorý sme dostali pribalený spolu s telefónom, alebo taktiež je často voľne dostupný na web stránkach výrobcu telefónu.

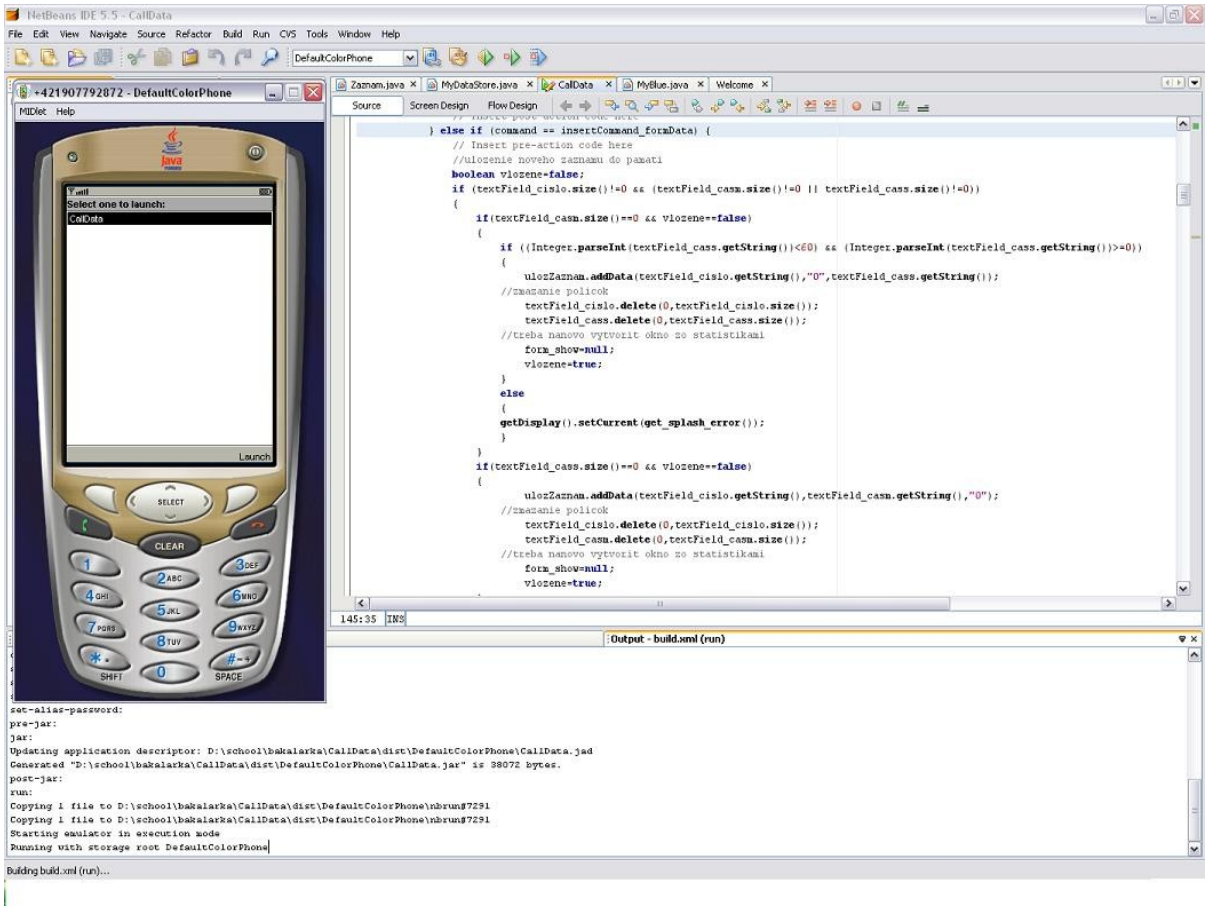
> Jedným z najdôležitejších nástrojov je emulátor. Umožňuje nám ladiť a testovať nami vyvíjanú aplikáciu priamo vo vývojovom prostredí, takže ju nemusíme neustále nahrávať na telefón. Do prostredia NetBeans sa dá doinštalovať emulátor *Sun Java Wireless Toolkit.*

Tento univerzálny emulátor má v základe vytvorené 4 profily telefónov, ďalšie sa dajú bez väčších problémov vytvoriť.

Emulátor okrem schopnosti samotného behu aplikácie obsahuje aj rôzne nástroje na ladenie ako je monitor pamäte alebo siete. Pomocou týchto nástrojov sa dá zistiť ako bude naša aplikácia

zaobchádzať s pamäťou reálneho zariadenia alebo môžme zistiť či máme správne inicializované sieťové spojenie.

vývojove prostredie NetBeans a emulátor Sun Java Wireless Toolkit



2.2 Návrh aplikácie

Mojou úlohou je vytvoriť jednoduchý informačný systém na evidenciu hovorov. Do systému sa po ukončení hovoru zapíše telefónne číslo na ktoré bol hovor uskutočnený a čas trvania hovoru.

Systém bude schopný na základe dátumu vloženia spočítať počet hovorov pre každý deň a taktiež spočítať celkovú dĺžku hovoru za deň.

Aplikácia bude taktiež obsahovať Bluetooh server, pomocou ktorého sa dáta budú môcť preniesť na PC alebo PDA. Na tento prenos budú určené dáta uložené v pamäti, nie zobrazená štatistika. Tieto dáta budú obsahovať záznam pre každý hovor. Udržované údaje budú pozostávať z telefónneho čísla, na ktoré bol hovor uskutočnený, dátumu a dĺžky hovoru.

3 Implementácia

Za jazyk implementácie som si zvolil J2ME, vzhľadom na jeho rozšírenosť medzi výrobcami telefónov. Tento jazyk má obrovskú výhodu vo svojej prenositeľnosti. Viac o tomto jazyku sa dozviete v kapitole *Úvod*.

Pri implementácií som využil viacero tried jazyka J2ME. Tieto triedy sú kvalitne zdokumentované v manuáli, ktorý sa nachádza voľne dostupný na stránkach spoločnosti SUN, ale aj v prostredí NetBeans. Po nastavení kurzoru na názov triedy nám NetBeans ponúkne zobraziť dokumentáciu k tejto triede, čo veľmi uľahčuje orientáciu v nej.

Na ukladanie dát v telefóne využívam triedu *RecordStore.*

3.1 Record Managment Store

Record managment store (RMS) pozostáva z kolekcie záznamov, ktoré sú trvale uchovávané, aj po ukončení aplikácie. Preto sú vhodné na ukladanie dát, ktoré chceme trvale uschovať, až do ich ručného zmazania. Je však možné, že pri niektorých typoch telefónov dáta nemusia prežiť po vybratí batérie. Každá aplikácia má možnosť vytvárať neobmedzený počet záznamov, každý však musí mať jednoznačný identifikátor.[1] Pri odstránení MIDLetu, dôjde aj k odstráneniu s ním spojených záznamov. Zdieľanie záznamov medzi aplikáciami nie je podporované, každá aplikácia môže využívať iba ten, ktorý sama vytvorila. To znamená, že dve rôzne aplikácie môžu mať záznam s rovnakým identifikátorom. O správu záznamov sa stará konkrétna implementácia Javy v telefóne. U väčšiny telefónov môže mať názov až 32 znakov a názov je *case-sensitive.*

V RMS nie sú implementované žiadne možnosti na uzamknutie prístupu k dátam. Preto ak používame vlákna a tieto pristupujú k dátam, je potrebné zaistiť ich vzájomnú koordináciu, aby nedošlo k strate alebo neželanému pozmeneniu dát.

Každý *Record Store* môže obsahovať neobmedzený počet záznamov.[2] K týmto záznamom má prístup pomocou unikátnych *ID*, ktoré začínaju od 1. Prvému záznamu je teda pridelené *ID=1,* nasledujúcemu o 1 väčšie. Ak niektorý záznam zmažeme, jeho *ID* v tomto *Record Store* už požité nebude.

Pre prácu s *RMS* je v J2ME určená trieda *javax.microedition.rms.* V mojej implementácií na využívam *RMS* na ukladanie záznamov o hovore do pamäte. Vytvoril som na to triedu *MyDataStore*,

1 obmedzenie predstavuje iba pamät telefónu

2 obmedzené veľkosťou typu *integer* na danej platforme

ktorá obsahuje metódy na prácu s *RMS* a v ňom uloženými záznamami. Údaje o hovore ukladám do pamäte pomocou metódy *addData.* Táto metóda po každom zavolaní s príslušnými parametrami vloží do pripraveného *Record Storu* jeden záznam v tvare:

```
telefónne_číslo@dátum@minúty:sekundy@
```

Každý zo záznamov má svoj identifikátor, pomocou ktorého sa dá naň odkazovať.

Trieda ďalej obsahuje metódy, ktoré vyhodnocujú záznamy a vytvárajú z nich štatistiky. Za týmto účelom bola vytvorená metóda *makeStatistics,* ktorá prechádza postupne celý *Record Store* a na základe dátumu roztrieďuje záznamy a počíta v ktorý deň bolo uskutočnených koľko hovorov a aká bola ich celková dĺžka trvania.

Po vyhodnotení sa štatistiky uložia do vektoru, ten sa odošle triede, ktorá ma na starosti vykresľovanie okien a tá tieto štatistiky vykreslí na displej. Z dôvodu úspory miesta na displeji a tiež rozmermi displejov mobilných telefónov sa štatistiky vypisujú v zjednodušenej podobe, ktorá neobsahuje telefónne čísla, na ktoré boli hovory uskutočnené. Výpis má nasledovný tvar:

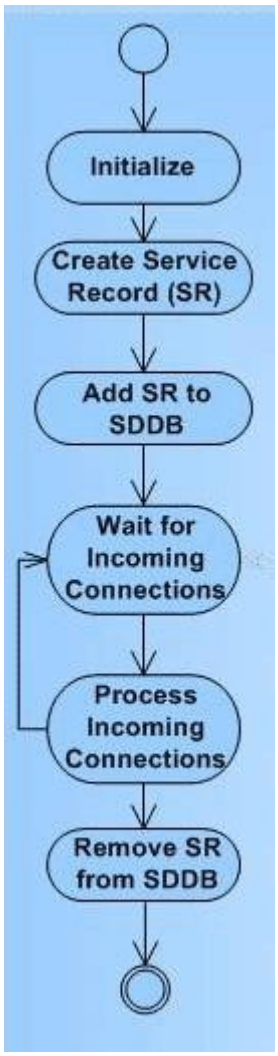
```
dátum|počet hovorov v tento deň|celková dĺžka trvania
```

Pri práci s dátami je občas potrebné ich premazať. Túto funkciu zastrešuje metóda *deleteStore,* ktorá po zavolaní z užívateľského menu zmaže všetky záznamy v *Record Store.*

Táto trieda, okrem metód na prácu s *RMS*, obsahuje tiež pomocné metódu na úpravu časových hodnôt, ktoré vzniknú pri spracovávaní štatistík. Metóda *myTime* je volaná z metódy *makeStatistics* a prevádza sekundy na minúty a minúty na hodiny. Pri vstupe 6 minút a 73 sekúnd, metóda vráti upravenú hodnotu a to 7 minút a 13 sekúnd. Metóda vracia čas vo forme reťazca, ktorý je rovno pripravený na zobrazenie na displeji. Medzi číselné hodnoty vsúva oddeľovače hodín, minút a sekúnd.

3.2 Bluetooth

Implementáciu technológie Bluetooth v J2ME zastrešuje trieda *javax.bluetooth*. V mojej aplikácií je na prácu s Bluetooth pripravená trieda *MyBlue*. Trieda pozostáva z metód určených na inicializáciu, posielanie a prijem správ. Aplikácia je vytvorená modelom *klient/*server, pričom sama zastupuje stranu serveru. Na obrázku sú znázornené aktivity, ktoré server vykonáva.



Inicializácia

Zariadenie, či už to je na strane klienta alebo serveru potrebuje pre svoju činnosť inicializáciu. Keďže aplikácia sa správa ako server, potrebuje službu, ktorú ponúka zaregistrovať, tak aby bola viditeľná a prístupná pre klienta, ktorý sa pripojí. Službu je teda potrebné pridať do *Service Discovery Database (SDDB)*.

Tiež je potrebné náš server prepnúť do módu kedy je viditeľný pre klienta. Na tento účel slúži metóda *setDiscoverable(int mode)*. Metóda má jeden parameter typu *int*, ktorým nastavujeme, viditeľnosť serveru. Máme dve možnosti:

- *DiscoveryAgent.LIAC:* zariadenie bude viditeľné iba na obmedzenú časovú dobu, typicky jednu minútu. Po tomto čase sa zariadenie samo prepne do neviditeľného režimu

- *DiscoveryAgent.GIAC:* zariadenie sa nastaví do viditeľného režimu bez časového obmedzenia. Tento režim využíva aj moja aplikácia.

Po zavolaní tejto metódy a uvedení zariadenia do viditeľného režimu je naša aplikácia viditeľná pre klienta, čím sa ale tiež môže stať potencionálnym cieľom útoku. Preto treba zvážiť, ktorý z režimov zvoliť. Moja aplikácia využíva prvý menovaný obmedzený režim *LIAC,* takže zariadenie je viditeľné iba obmedzenú dobu po inicializácií.

Vytvorenie služby

Implementácia Bluetoothu automaticky vytvorí záznam o službe po zavolaní metódy *StreamConnectionNotifier.*

Každá Bluetooth služba má svoj jednoznačný identifikátor *Universally Unique Identifier (UUID).* Tento identifikátor musíme prideliť každej službe, ktorú chceme využívať. Na vytvorenie sa využíva trieda *UUID,* pomocou ktorej môžme vytvoriť krátke (16-bitové alebo 32 bitové*)* alebo dlhé (128-bitové) *UUID* .

Vytvorenie spojenia

Bluetooth spojenie pomocou triedy *javax.microedition.io.Connector.* V tejto triede sa nachádza metóda *open()*, ktorej pridáme ako parameter *URL* spojenia. V mojej aplikácií využívam spojenie pomocou protokolu *RFCOMM* a spojenie, ktoré zabezpečuje trieda *StreamConnection. URL* má teda tvar:

```
btspp://localhost:RFCOMM_UUID +name=CallData2007FIT;authenticate=false
```

kde

- *btspp:* je schéma pre *RFCOMM*
- *localhost:* je *hostname, localhost* sa používa pre stranu servera, pri klientskej aplikácií sa udáva adresa zariadenia, na ktorom beží server
- *RFCOM_UUID:* je identifikátor služby, ktorý sa využíva na nastavenie služby. Podľa tohto identifikátoru klient vyhľadá službu na servery.
- *name=CallData2007FIT:* je názov služby
- *authenticate=false:* vypína overovanie

Po tom čo zostavím *URL,* môžem server spustiť. Na spustenie serveru a čakanie na klienta slúži metóda *acceptAndOpen().* Po tom čo sa klient pripojí, metóda vráti spojenie typu *StreamConnection.*

```
StreamConnection connection = StreamConnectionNotifier.acceptAndOpen();
```

Keď máme takto vytvoré spojenie, môžme získať informácie o pripojenom zariadení pomocou metód obsiahnutých v triede *RemoteDevice.*

```
RemoteDevice remoteD = RemoteDevice.getRemoteDevice(connection);
```

Odosielanie informácií

Na odosielanie informácií využívam triedu *OutputStream* a jej metódu *write*. Tejto metóde predávam reťazec, ktorý sa má odoslať. Najskôr pošlem dĺžku posielaného reťazca, potom reťazec samotný. Na jedno odoslanie sa pošle jeden záznam o hovore. Celý proces ukončím poslaním ukončovacieho reťazca (@@@). Tým dám klientovi správu, že už obdržal všetky záznamy.

Prehľad o službe

Atribút	Hodnota
UUID	1234500000100080000805F9B34FB
Názov protokolu	RFCOMM
Meno služby	CallData2007FIT
URL	`btspp://localhost:RFCOMM_UUID+name=CallData2007FIT;authenticate=false`

4 Záver

Cieľom mojej práce bolo vytvoriť aplikáciu, ktorá slúži ako jednoduchý informačný systém pre mobilný telefón a umožňuje evidenciu hovorov, ktoré boli uskutočnené z daného zariadenia. Aplikácia taktiež vyhodnocuje zhromaždené údaje a vytvára štatistiky.

Vzhľadom na rozšírenosť medzi výrobcami mobilných telefónov som za implementačný jazyk zvolil Java 2 Micro Edition (J2ME). Voľba prostredia vhodného na vývoj spadla na NetBeans. Jeho výhodou je, že obsahuje zásuvný modul, ktorý podporuje vývoj mobilných aplikácií. Tento modul sa volá *NetBeans Mobility Pack.* Ako emulátor som použil *Sun Java Wireless Toolkit.*

Počas vývoja som sa stretol s viacerými problémami. Pri ukladaní dát do *Record Managment Store* a behu aplikácie na emulátore bolo problematické sledovať fyzické ukladanie dát a to či sa ukladajú v správnom tvare. Vzhľadom k tomu, že dáta sa ukladajú natrvalo a zostávajú v pamäti aj po vypnutí telefónu (emulátoru), na ich ručne mazanie bolo treba využívať funkcie, ktoré nám *Sun Java Wireless Toolkit* ponúka.

Problematický je taktiež vývoj Bluetooth serveru na emulátore. Ten je síce natoľko funkčný, že dokáže spoznať bežiacu aplikáciu na druhom, nezávislom emulátore, ale napriek tomu bolo treba paralelne zostaviť druhú, testovaciu aplikáciu, pomocou ktorej som môj server ladil. Ako táto testovacia aplikácia mi po malej úprave poslúžil jeden z ukážkových programov, ktoré sú dostupné spolu s emulátorom.

Mobilný telefón, tak ako väčšina malých zariadení nedisponuje tak veľkou operačnou pamäťou akú máme k dispozícii pri programovaní pre PC. Preto je obzvlášť potrebné pracovať s pamäťou tak, aby jej moja aplikácia spotrebovávala čo najmenej a aby nepotrebná pamäť bola v poriadku uvoľňovaná. K tomuto cieľu je vhodné využívať *garbage collector*, ktorý je súčasťou J2ME.

Nevýhodou mojej implementácie je to, že užívateľ musí ručne zapisovať údaje cez dostupný formulár. Toto riešenie je však nevyhnutné, vzhľadom na to, že jazyk J2ME neobsahuje žiadne metódy na zachytávanie odchádzajúcich hovorov a ani prístup do denníka hovorov, ktorý sa v každom telefóne nachádza. Túto možnosť poskytujú až telefóny vyššej generácie, ktoré majú v sebe implementovaný operačný systém Symbian alebo Windows Mobile.

Ďalší vývoj aplikácie by mal smerovať na automatizáciu, čiže aby sa dáta do pamäte ukladali automaticky a užívateľ by nemusel nič manuálne po každom hovore vypisovať. Táto funkcia je však veľmi závislá na mobilnom telefóne a platforme, na ktorej telefón beží.

Literatúra

[1] Qusay H. Mahmoud Naučte se Java 2 Micro Edition, ISBN 80-247-0444-7

[2] Symbian OS the mobile operating system, www.symbian.com, máj2007

[3] Wikipedia The free encyklopedia, en.wikipedia.org, máj 2007

[4] Wireless Application Programming with J2ME and Bluetooth, developers.sun.com, máj 2007

[5] Using the Java APIs for Bluetooth, developers.sun.com, apríl 2007

Zoznam príloh

1. CD
    - zdrojové kódy aplikacie
    - manuál
    - technická správa v elektronickej podobe