VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

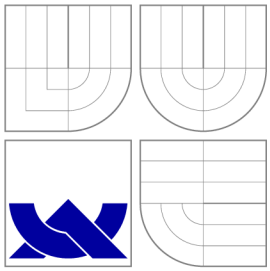
DÁLKOVÉ OVLÁDÁNÍ POČÍTAČE POMOCÍ MOBILNÍHO TELEFONU

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

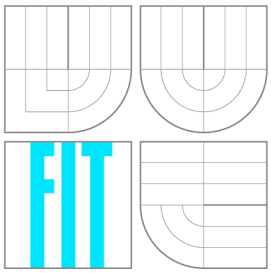
AUTOR PRÁCE
AUTHOR

JAN KOLAŘÍK

BRNO 2010



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

DÁLKOVÉ OVLÁDÁNÍ POČÍTAČE POMOCÍ MOBILNÍHO TELEFONU

COMPUTER REMOTE CONTROL USING CELL PHONE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

JAN KOLAŘÍK

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. PETR POSPÍCHAL

BRNO 2010

Abstrakt

Výsledkem mé práce by měla být aplikace, která umožňuje ovládání videí a hudby pomocí mobilního telefonu, který má operační systém Andriod 2.1.

Abstract

The result of my work would be an appilcation, which enables to control videos and music by a cell phone, which has an operating system Android 2.1.

Klíčová slova

Android, Java, Dálkové ovládání, Linux, Bluetooth.

Keywords

Android, Java, Remote control, Linux, Bluetooth.

Citace

Jan Kolařík: Dálkové ovládání počítače pomocí mobilního telefonu, bakalářská práce, Brno, FIT VUT v Brně, 2010

Dálkové ovládání počítače pomocí mobilního telefonu

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Petra Pospíchala.

.....
Jan Kolařík
18. 5. 2011

Poděkování

Zde je možné uvést poděkování vedoucímu práce a těm, kteří poskytli odbornou pomoc.

Obsah

Kapitola 1

Úvod

V dnešní době moderních technologií a jejich rychlého rozvoje, se požadavky uživatelů, jak na hardware, tak software, neustále zvyšují. Příkladem za všechny je až neuvěřitelně rychlý rozvoj mobilních telefonů či smartphonů. Co je dnes novinkou, je zítra zastaralé. Skoro každý člověk vlastní mobilní telefon a je s ním sžitý tak, že ho stále nosí při sobě a tím pádem na něj klade čím dál větší nároky. V současnosti není problém z mobilního telefonu přistupovat na internet, provádět bankovní převody, komunikovat elektronicky se svým okolím. Pokud je telefon využíván k tolika úkonům, přichází také například požadavek ovládat s jeho pomocí z pohodlí postele přehrávání videa a hudby na PC.

V tento text bude pojednávat o vytvoření programu pro ovládání videa a hudby na PC pomocí mobilního telefonu. Existuje však velké množství různých mobilních telefonů i jejich operačních systémů. V této práci je jako cílový operační systém telefonu zvolen Android [21], který je výtvorem firmy Google. Aplikace pro něj se vyvíjejí hlavně v Javě, s použitím SDK (Software Development Kit) od výrobce.[16] Hlavními přednostmi jazyka Java je preciznost OOP návrhu, jeho přenositelnost a jednoduchost. Aplikace bude založená na principu klient - server. Obě strany spolu budou komunikovat pomocí technologie Bluetooth.

Kapitola 2

Technologie Bluetooth

Bluetooth je bezdrátová komunikační technologie, která slouží pro propojení a výměnu dat mezi dvěma a více elektronickými zařízeními na krátkou vzdálenost. Používá se například u osobních počítačů, PDA, mobilních telefonů, GPS a dalších. Původním účelem technologie Bluetooth bylo nahradit spojování dvou zařízení kabelem. [9]

2.1 Historie

Slovo Bluetooth pochází z poangličtěného jména dánského krále Haralda Modrozubého - Harald Bluetooth, který vládl v 10. století a sjednotil Skandinávii. Byl to dobrý vyjednavač, a právě z této jeho vlastnosti vychází ideologie Bluetooth, která je schopná zajistit spojení i mezi diametrálně odlišnými zařízeními. Dalším důvodem výběru jména Bluetooth pro tuto technologii byl fakt, že první zařízení bylo vyvíjeno právě ve Skandinávii. Při tvorbě loga pro Bluetooth tvůrci vycházeli ze stejného základu. Iniciály H.B. byly přepsány do run a z nich se stalo originální logo, které je na obrázku 2.1.[9, 24]

Obrázek 2.1: Logo Bluetooth.[2]

Úplné počátky technologie Bluetooth spadají již do roku 1994, kdy divize firmy Ericsson vytvořila studii o možnosti náhrady současného propojovacího kabelu RS-232 mezi zařízeními. Následně byla v roce 1998 založena pěti firmami (Ericsson, IBM, Intel, Nokia a Toshiba) skupina Bluetooth Special Interest Group (BSIG, nebo Bluetooth SIG). Tato poměrně malá skupina firem se brzy rozrostla o 3Com, Lucent, Microsoft a Motorolu, ale pořád přicházeli noví a noví členové. V současti má tato skupina už více než 10 000 členů. Díky nízké ceně a nízkým energetickým nárokům na provoz si Bluetooth oblíbili jak uživatelé, tak i výrobci elektroniky. [2, 24]

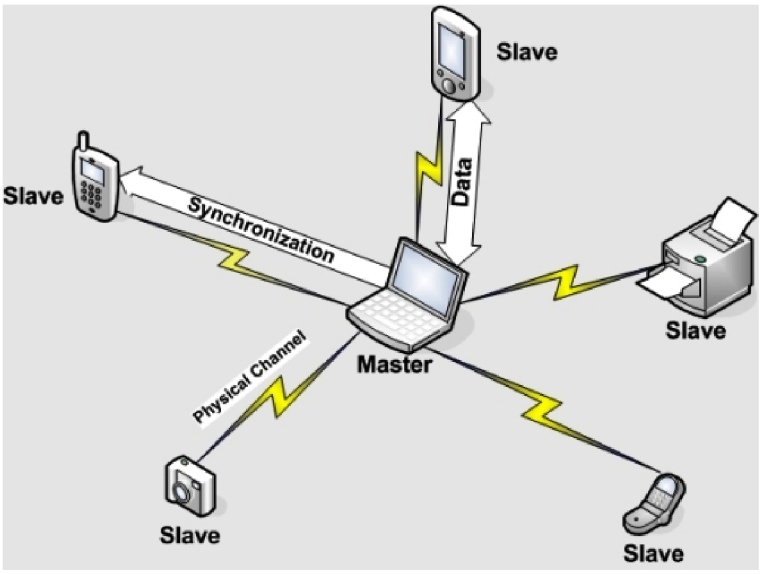
2.2 PAN

PAN je zkratka z anglického slovního spojeni *Personal Area Network*, což volně přeloženo do češtiny znamená osobní síť. PAN je tvořena komunikačními zařízeními, jako notebook, PDA, mobilní telefon, které jsou v blízkosti jedné osoby. Dosah těchto sítí bývá několik metrů. PAN se používá pro přenos dat mezi jednotlivými zařízeními, nebo pro připojení k okolní síti či Internetu. [10]

PAN sítě mohou být:

- Drátové - například pomocí USB, FireWire ...
- Bezdrátové - pomocí IrDA, Bluetooth ...

PAN se také někdy označuje jako piconet. Specifikace a počty zařízení v piconetu budou popsány v následující sekci. Příklad toho, jak vypadá PAN je uveden na obrázku 2.2.



Obrázek 2.2: Ilustrativní schéma PAN. [19]

2.3 Specifikace

Technologie Bluetooth je definována standartem IEEE 802.15.1 a spadá do kategorie osobních počítačových sítí PAN. Pracuje v ISM (Industrial, Scientific and Medical) pásmu 2,4 GHz (stejně jako Wi-Fi). Bluetooth používá bezdrátovou technologii zvanou Frequency Hopping Spread Spectrum, která v průběhu jedné sekundy provede 1600 přeladění mezi 79 frekvencemi s rozestupem 1MHz. Tento mechanizmus je zaveden z důvodu zvýšení odolnosti spojení vůči rušení na stejné frekvenci. Maximální vzdálenost spojení zařízení je 100 m, ale toto platí pro ideální podmínky (volný prostor). Ve skutečnosti je maximální dosah ovlivňován pevnými překážkami mezi zařízeními. Bluetooth podporuje přenos jak dat, tak i hlasu. Bluetooth zařízení jsou rozdělena do jednotlivých tříd podle vysílacího výkonu, viz. tabulka 2.1. [24, 9]

Třída Class 1 se používá převážně u osobních počítačů, třída Class 2 pak většinou pro mobilní telefony. Třída Class 3 se v dnešní době nevyužívá.

Třída	Maximální povolený výkon		Dosah
	mW	dBm	
Class 1	100	20	100 m
Class 2	2,5	4	10 m
Class 3	1	0	1 m

Tabulka 2.1: Třídy Bluetooth.

Bluetooth zařízení může pracovat ve dvou režimech, master a slave. Master je řídící zařízení a může obsluhovat ad-hoc[1] až sedm zařízení v režimu slave. V tomto případě se jedná o mnohabodovou komunikaci (multipoint) a zařízení se nachází v piconetu. Pokud je k masteru připojeno pouze jedno slave zařízení, jde o dvoubodovou (point-to-point) komunikaci.

Standart Bluetooth podporuje dva způsoby komunikace.

- Synchronní spojovaná komunikace (Synchronous Connection Oriented Link – SCO) realizuje dvoubodovou (point-to-point) komunikaci mezi zařízením typu master a právě jedním zařízením typu slave. Zařízení master posílá v zadaných časových intervalech data na zařízení slave. Slave může mezi jednotlivými daty odeslat svoje data. Master může realizovat až tři spojení typu SCO, a to buď s jedním nebo více slave. SCO je primárně určeno pro přenos hlasu. V případě ztráty nebo chyby packetu se nežádá o jeho opětovné zaslání, ale přeskočí se. Aby se při zhoršení podmínek udržela chybovost v přijatelných mezích, byla pro tento účel vyvinuta metoda FEC (Forward Error Connection), která vkládá do přenosu redundantní data a s jejich pomocí je schopná najít a opravit určitý počet chyb.

- Asynchronní nespojovaná komunikace (Asynchronous Connectionless Link – ACL) se používá pro komunikaci mezi zařízením typu master a jedním nebo více zařízeními typu slave, pouze tehdy když není kanál rezervován SCO. Mezi zařízeními master a slave může být v jeden okamžik pouze jedno spojení ACL. ACL je na rozdíl od SCO navrženo pro spolehlivý přenos dat, tzn. že v případě ztráty packetu je znovu poslán a na časovou prodlevu při přeposílání dat není kladen tak velký důraz.

Každé zařízení má svoji unikátní 48 bitovou adresu BD_ADDR (Bluetooth Device Address).[24, 9]

2.4 Verze

V této sekci se budeme zabývat jednotlivými verzemi Bluetooth(1.0, 1.0B, 1.1, 1.2, 2.0+EDR, 2.1+EDR, 3.0 a následující verzí 4.0)[2] Bluetooth prošlo od roku 1998 značným vývojem.

- **1.0, 1.0B**
  Tyto verze se příliš neuchytily, protože trpěly spoustou nedostatků. Největším problémem byla nízká schopnost komunikace s jinými zařízeními.

- **1.1**
  Až s nástupem této verze se dá mluvit o rozmachu Bluetooth, opravuje spoustu chyb z verze 1.0B a je popsána standardem IEEE_802.15. Zásadním zlepšením je zvýšení interoperability mezi zařízeními.

[1] Spojení peer-to-peer používané mezi zařízeními.

- **1.2**
  Tato verze je instalována do většiny mobilních telefonů. Je kompatibilní s verzí 1.1. Podporuje Adaptive Frequency Hopping, které slouží k omezení rušení tím, že nepoužívá přeplněné frekvence při přeladďování. Pracuje na přenosové rychlosti 1 Mib/s. Zvyšuje kvalitu přenášeného audia pomocí detekce a opravy chyb. Obsahuje indikátor úrovně signálu. Přibyla možnost Fast Connection and Discovery, sloužící k rychlému připojení a vyhledávání okolních zařízení.

- **2.0 + EDR**
  Verze byla ohlášena SIG v roce 2004 a začala se používat v roce 2005. Je zpětně kompatibilní s verzí 1.2 a umožňuje trojnásobnou přenosovou rychlost (3 Mib/s). Zlepšuje komunikaci typu multipoint a také má nižší spotřebu energie. Podporuje komunikace typu broadcast a multicast.

- **2.1 + EDR**
  Vylepšuje párování zařízení a optimalizuje spotřebu energie.

- **3.0 + HS**
  Tato verze se začala používat v roce 2009 a je schopna pracovat na teoretické rychlosti až 24 Mib/s.

2.5 Stack

Protokol Stack [24, 23, 2] byl specifikován skupinou SIG. Hlavním důvodem této specifikace je určit protokol, kterým se musí výrobci zařízení Bluetooth řídit, aby byli schopni komunikovat se zařízeními jiných výrobců. Protokol stack rozděluje Bluetooth do jednotlivých vrstev, podobně jako ISO/OSI u ethernetu. Samotné schéma stacku je zobrazeno na obrázku 2.3. Aby aplikace mohly komunikovat mezi sebou, musí mít stejnou verzi protokolu stack.

Následuje popis jednotlivých vrstev:

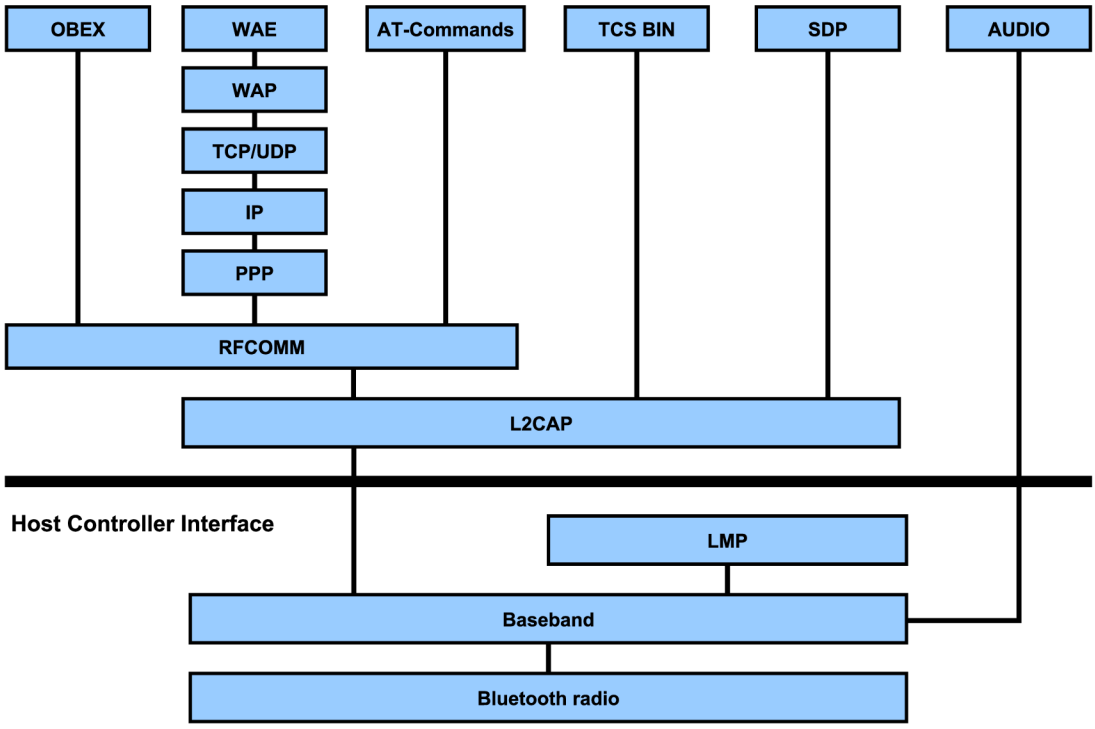
- PPP
  Komunikační protokol využívaný pro vytvoření přímého spojení mezi dvěma síťovými uzly.

- WAE/WAP
  WAE specifikuje základní kostru pro bezdrátová zařízení. WAP slouží pro přístup k internetu.

- TCP/IP/UDP
  Základní protokoly sloužící pro přenos informací v síti internet.

- Object Exchange Protocol (OBEX)
  Jde o přenosový protokol, definující datové objekty, a komunikační protokol dvou zařízení. Zařízení si mohou objekty vyměňovat. S ohledem na vysokou podobnost mezi nižšími vrstvami IrDA a Bluetooth protocol stacku, převzalo Bluetooth OBEX z IrDA.

- AT – Commands
  Obsahuje definice signálů pro sériové modemy.



Obrázek 2.3: Bluetooth stack. [24]

- Telephone Control Protocol (TCS BIN)
  Definuje řízení signálů při hlasových a datových přenosech mezi Bluetooth zařízeními.
- Audio
  Vrstva poskytující služby pro přenos zvuku. Je nezávislá na L2CAP vrstvě.
- Radio Frequency Communication Protocol (RFCOMM)
  Slouží jako náhrada sériového kabelu. Emulací řízení RS-232 signálů přes fyzickou vrstvu (Baseband layer) vytváří sériový datový proud.
- Service Discovery Protocol (SDP)
  Umožňuje vyhledat, které služby zařízení podporuje, jaké parametry jsou pro dané připojení nutné. Např.: Server je vyhledán klientským zařízením, které zjišťuje, jaké profily tento server podporuje.
- Logical Link Control & Adaptation Protocol (L2CAP)
  Vrstva používaná vyššími vrstvami pro spojové a nespojové přenosy. Jejím úkolem je segmentace a sestavování packetů. Poskytuje spolehlivé spojení a „flow control“.
- Host Controller Interface
  Slouží jako rozhraní mezi nižšími a vyššími vrstvami protokolu stack.
- Link Manager Protocol
  Úkolem této vrstvy je navazování a udržování spojení mezi zařízeními. Provádí také řízení komunikace a je zodpovědná za výměnu zpráv týkajících se bezpečnosti (autentizace, výměna šifrovacích klíčů).

- Baseband
  Zajišťuje formátování dat pro přenos z Bluetooth radio vrstvy a zpět na ni. Zároveň se stará o synchronizaci spojení.

- Bluetooth radio
  Protokol nejnižší vrstvy. Jeho úkolem je modulace a demodulace dat na signály pro přenos vzduchem. Popisuje fyzické požadavky na přijímač a vysílač.

2.6 Profily

Profily [23, 8, 24]slouží k definování různých činností na zařízení Bluetooth a komunikaci zařízení mezi sebou. Profily poskytují kompatibilitu mezi různými zařízeními, které ale musí používat stejný profil.

Každý profil musí obsahovat následující specifikace:

- Závislosti na jiných profilech
- Formát uživatelského rozhraní
- Části Bluetooth protokol stacku využívané daným profilem

Charakteristika nejčastějších profilů:

- Generic Acces Profile
  Základní profil, který podporují všechna zařízení. Jeho úkolem je zajištění vyhledávání a vytváření spojení mezi zařízeními.

- Generic Object Exchange
  Jde o základní profil pro všechny aplikace využívající OBEX. Provádí přenosy objektů mezi zařízeními, synchronizaci a přenos dat.

- Serial Port Profile
  Tento profil slouží jako náhrada za sériový kabel. Využívá zařízení používající Bluetooth ke komunikaci se sériovým portem. Popisuje postup emulace sériového portu a způsob připojení k Bluetooth zařízení.

- File Tranfer Profile
  Definuje na vzdáleném zažízení postupy, jak provádět přenos dat a jak s nimi manipulovat. Tento profil také umožňuje přístup k souborovému systému na vzdáleném Bluetooth zařízení.

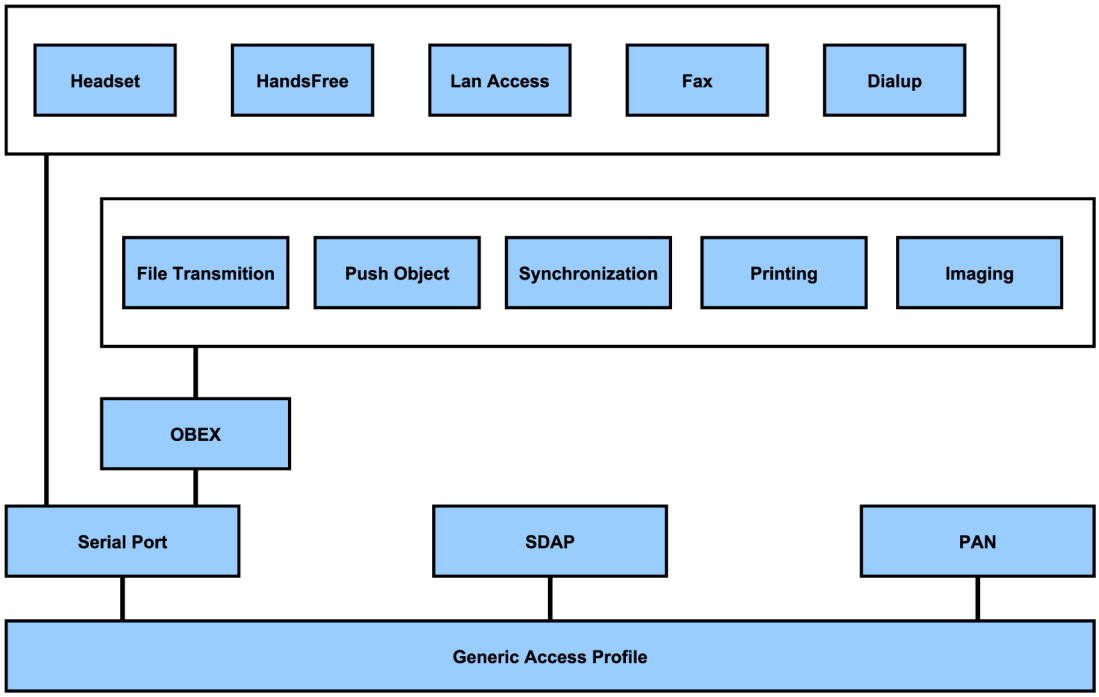
- Service Discovery Application Profile
  Obsahuje popis, jak použít Service Discovery Protocol za účelem nalezení služby na vzdáleném zařízení.

- Personal Area Networking
  Definuje způsob, jakým mohou dvě nebo více zařízení vytvořit ad-hoc síť a jak se připojit ke vzdálené síti přes přístupový bod.

- Object Push Profile
  Definuje postup, jak přenášet objekty mezi zařízeními.

- Synchronization Profile
  Slouží k synchronizaci PIM (Personel Information Manager) položek mezi aplikacemi, např. adresáře nebo kalendáře.
- HandsFree Profile
  Popisuje připojení HandsFree sady k zařízením podporujícím Bluetooth.
- Headset Profile
  Profil vytvořený pro přenos zvuku s pomocí Bluetooth. Konkrétně popisuje připojení sluchátek a mikrofonu.
- DialUp Network Profile
  Standard pro výtačené připojení na internet realizované pomocí modemu. Nejčastěji jde o využití pro připojení počítače k internetu přes mobilní telefon.

Hierarchie nejpoužívanějších profilů Bluetooth je uvedena na obrázku 2.4.



Obrázek 2.4: Hierarchie profilů. [24]

2.7 Srovnání bezdrátových technologií

Tři bezdrátové technologie, se kterými se můžeme běžně setkat, jsou uvedeny v tabulce 2.2. IrDA [13] (Infrared Data Association) přijímá a vysílá infračervené světlo, které je emitováno LED diodou, nebo infračervenou laserovou diodou. Slouží pro propojení právě dvou zařízení (PC a telefon, telefon a telefon) na krátkou vzdálenost, a spojení je podmíněno přímou viditelností obou zařízení. Wi–Fi [14] (wireless fidelity) se používá především pro připojení počítačů, notebooků a jiných zařízení podporujících tento standard do lokální sítě.

Oproti IrDA má velký dosah, který lze v případě, že to zařízení podporuje, zvětšit použitím externí antény. I jeho rychlosti jsou vyšší než u IrDA a nevyžaduje přímou viditelnost mezi zařízeními pro přenos. Technologie Bluetooth je popsána výše. Má malé energetické nároky, dosah několik metrů a do budoucna se jeví jako velmi perspektivní technologie v mnoha oblastech.

Technologie	IrDa	Wi–Fi	BlueTooth
Způsob přenosu	Opticky (vlnová délka 875 nm)	Radiová frekvence (2,4 GHz a 5 GHz)	Radiová frekvence (2,4 GHz)
Maximální přenosová rychlost	300 KiB/s - 14 MiB/s	až 600 Mib/s (802.11n)	až 24 Mib/s
Dosah	1 m	100 m (bez externí antény)	100 m
Počet komunikujících zařízení	2	Neozmeno	8 aktivních, 200 pasivních
Adresování	32-bitové ID	48-bitová MAC	48-bitová MAC
Hlasové kanály	Nepodporuje	VoIP	3

Tabulka 2.2: Srovnání bezdrátových technologií. [13, 14, 2]

Kapitola 3

Java

Podle Tiobe indexu patří Java mezi nejpopulárnější progrmovací jazyk.

3.1 Historie

Od roku 1991 se firma Sun Microsystems začala zabývat vývojem jazyka na principech C / C++ pro vestavěné systémy. Jazyk se měl původně jmenovat Oak (dub), podle stromu, který stál pod okny vedoucího týmu vývojářů. Ale jazyk tohoto jména již existoval, tak vývojová skupina začala hledat jiné vhodné jméno. Údajně po návštěvě firemního bufetu se rozhodli pro jméno Java, což znamená „kafé“. Projektu se moc nedařilo, dokud si v roce 1993 firma Sun neuvědomila vzrůstající potenciál WWW a možnosti programování aplikací pro něj. V květnu 1995 byla Java oficiálně představena na konferenci. Od tohoto okamžiku si Java začala získávat stále více příznivců a mnozí si také uvědomili, jaký potenciál skrývá, co by běžný programovací jazyk.[22]

3.2 Zpracování programu v Javě

Překlad v Javě neprobíhá do jazyka relativního (do .OBJ), ale do pseudojazyka nazývaného *byte-code*. Tento jazyk je nezávislý na platformě, takže programátor nemusí řešit odlišnosti programování pro jednotlivé platformy. Soubory přeloženého programu (do byte-codu) mají příponu *.class*. Při zavádění souborů do paměti probíhá jejich ověřování, které má za úkol dosáhnout velmi vysoké bezpečnosti spuštěného programu. Je tomu tak z důvodu ochrany toho, kdo program spouští. Po ověření je program spuštěn interpretem. Jedná se tedy o interpretovaný jazyk.[22]

3.3 Základní vlastnosti jazyka Java

V této sekci je uveden stručný seznam podstatných základních vlastností [7] jazyka Java.

- Jednoduchost – syntaxe jazyka vychází z C a C++, ale jsou odstraněny konstrukce, které způsobovaly programátorům problémy. Jsou ale přidána některá užitečná rozšíření (např. foreach).
- Objektová orientace – kromě osmi základních datových typů jsou všechny ostatní objektové.

- Distribuovatelnost – obsahuje podporu pro síťové aplikace.
- Robustnost – je navržen pro vývoj vysoce bezpečného softwaru, z tohoto důvodu neobsahuje jazyk konstrukce, ve kterých se často chybuje (např.: používání ukazatelů, příkaz goto . . .).
- Garbage collector – mechanizmus, který má za úkol hledat a uvolňovat již nepoužívané části paměti.
- Nezávislost na platformě – vytvořená aplikace poběží na libovolném operačním systému a architektuře, jedinou podmínkou běhu je přítomnost virtuálního stroje pro tuto platformu.
- Přenositelnost – nejenže není jazyk závislý na platformě, ale je i nezávislý co se týče elementárních datových typů. Přenositlenost je ovšem omezena pouze na stejnou edici Javy, pro kterou byla určena. U jiných edicí nemusí být dostupné použité objekty a jejich metody potřebné pro běh.
- Výkonnost – Přestože je Java jazykem interpretovaným, ztráta výkonu není nijak velká. Do strojového kódu se totiž překládá jen ten kód, který je skutečně potřeba.

3.4 J2SE

J2SE (Java 2 Standard Edition) je jednou ze čtyř základních edicí Javy. Dalšími edicemi jsou Java Card, která je určená pro běh na paměťových kartách dalších podobných zařízení s vlastní pamětí, jako jsou SIM karty, či kreditní karty. J2ME (Java 2 Micro Edition) je určená pro vestavěné systémy, například mobilní telefony nebo set-top boxy. Pro stolní počítače je určena J2SE a pro servery J2EE. Přehled edicí Javy je na obrázku 3.1.[7]

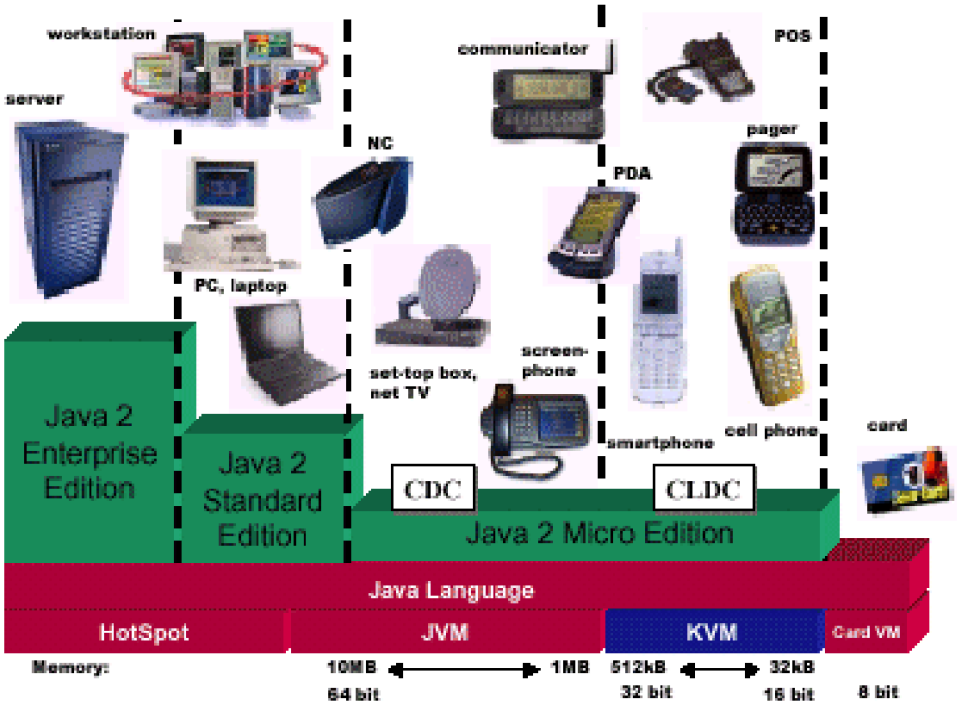
J2SE patří mezi nejrozšířenejší edici Javy a poskytuje všechny potřebné nástroje pro vývoj plnohodnotných přenositelných aplikací. Obsahuje i knihovny pro tvorbu uřivatelských grafických rozhraní (AWT, Swing). Dále obsahuje Java Debugger pro krokování programu, Javadoc pro snadné generování programové dokumentace, JDBC pro unifikovaný přístup k databázím. Samozřejmě nechybí ani přímá podpora sítí či XML. Úplný přehled struktury J2SE je na obrázku 3.2. [12]

3.5 Podpora Bluetooth

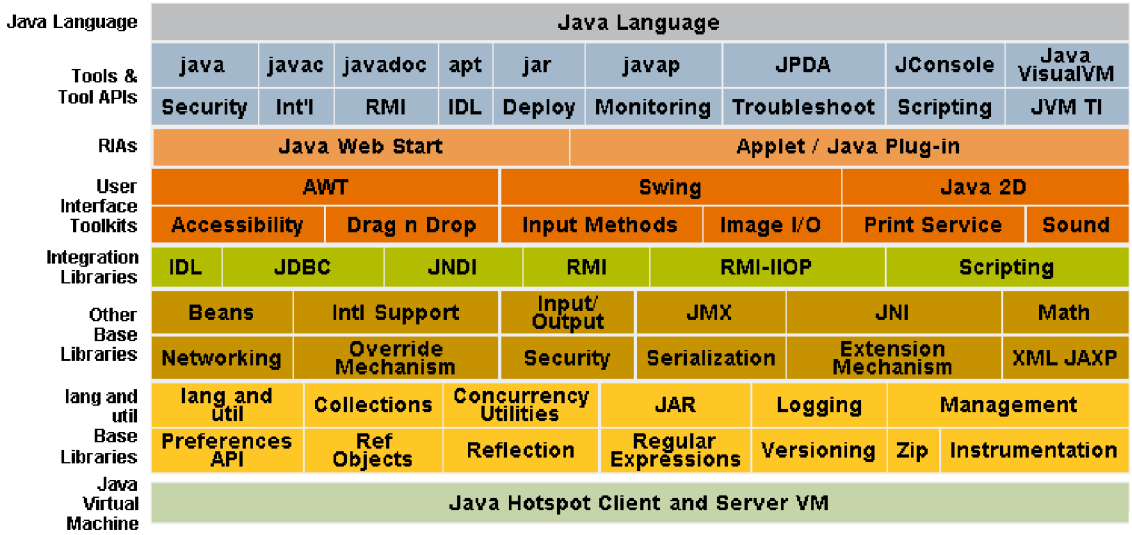
V dřívější době nebylo snadné vytvářet aplikace obsluhující Bluetooth. Programy byly nepřenositelné, jelikož používaly balíčky definované výrobcem. Tento problém odstranilo zavedení otevřeného standardu JSR-82. Při tvorbě této práce byla použita implementace *BlueCove*. Bluetooth stack je na obrázku 3.3. Nevýhodou této knihovny je fakt, že aby bylo zařízení viditelné pro svoje okolí, je pro toto nastavení nutné spouštět aplikaci jako root.[1]

JSR-82 ovšem nepodporuje všechny Bluetooth profily. Podporovány jsou:

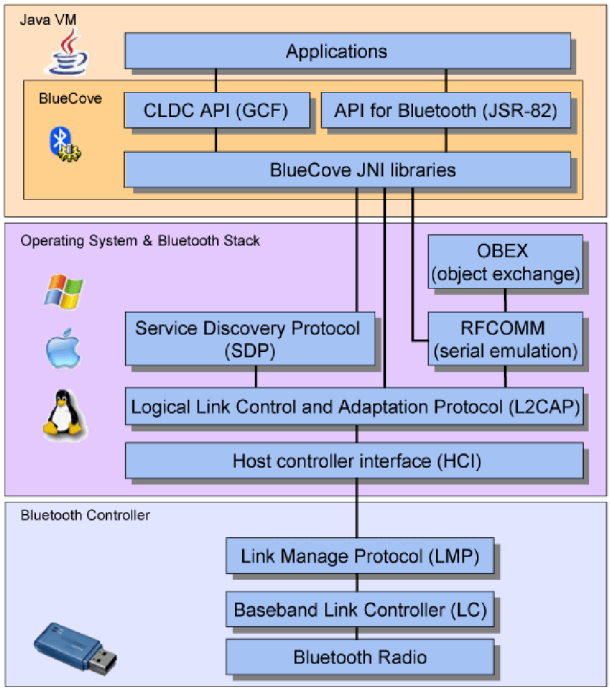
- L2CAP
- OBEX
- RFCOMM
- SDAP



Obrázek 3.1: Přehled edicí Javy. [20]



Obrázek 3.2: Struktura J2SE.[4]



Obrázek 3.3: Bluetooth stack s BlueCove. [1]

Kapitola 4

Android

4.1 Co je Android

Android je softwarová platforma založená na jádru Linuxu a využívá se na poli chytrých telefonů, PDA, tabletů a jiných. Základní revoluční myšlenkou této platformy je možnost stahovat si do svého zařízení nové aplikace přímo z internetu, pomocí programu Market, díky němuž může libovolný uživatel/programátor zpřístupnit svoje softwarová díla a dál je tak šířit. Nespornou výhodou pro moderního člověka je plná podpora spolupráce s asi nejmodernějším emailovým hostingem – gmail.com. Uživatel si může svoje kontakty a jejich detaily, poznámky či dokumenty ukládat přímo na gmail a v nejhorším případě, jako je krádež nebo ztráta zařízení, nemusí nutně přijít o všechna data. Samozřejmostí je také přímá podpora pro GPS, Wi-Fi či Bluetooth. Tento pokrok má ale pro uživatele i stinnou stránku. Tou je fakt, že pokud nedisponujete neomezeným připojením, budete odkázáni na příležitostné Wi-Fi spoty. Bez stálého připojení oceníte propracovanost Androidu asi jenom z poloviny.

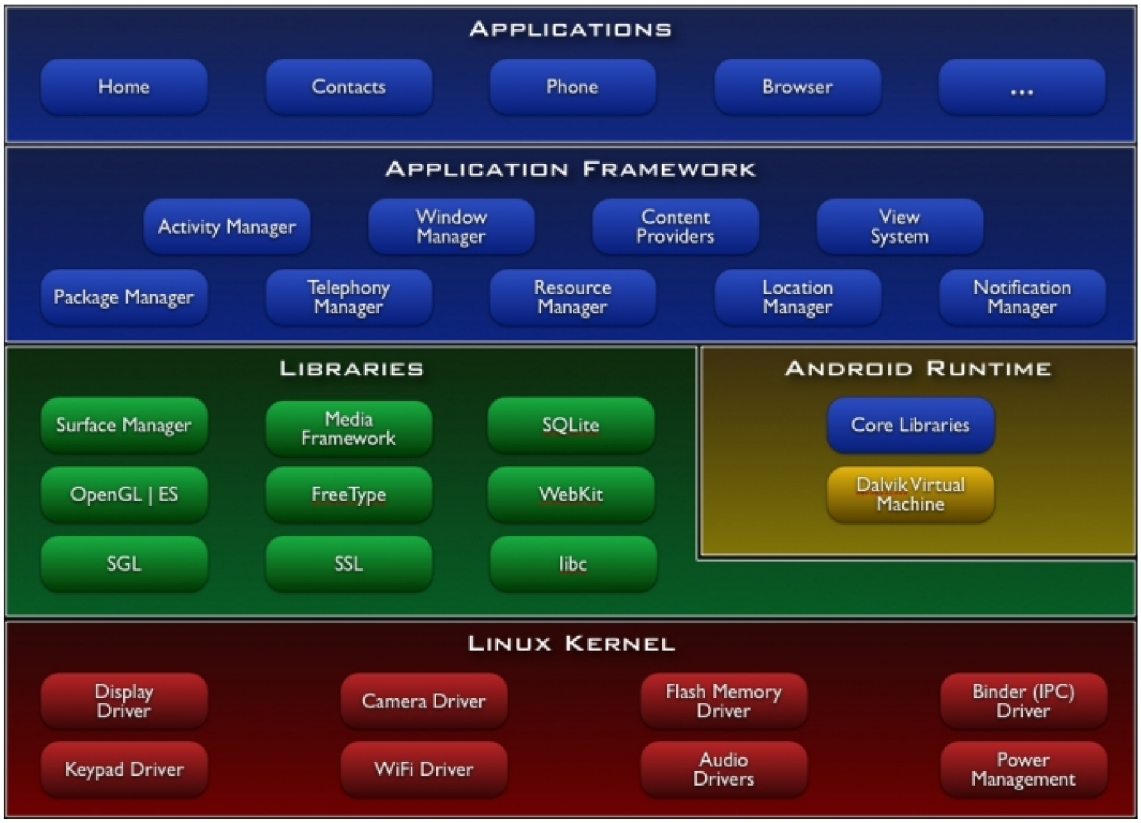
4.2 Historie

První verze spatřila světlo světa 5. listopadu 2007 a současně s ní bylo Googlem založeno OHA (Open Handset Alliance). Google nejenže založil tuto organizaci, ale také financoval soutěž Android Developer Challenge, ze které vznikly první aplikace. Již od roku 2008 je Android dostupný pro veřejnost, pod licencí Apache free-software and open-source license [11].

Obrázek 4.1: Logo Androida. [11]

4.3 Architektura Androida

Android může být nejlépe popsán pomocí vrstvového modelu.



Obrázek 4.2: Vrstvový model Androidu [21]

Nejnižší vrstva modelu (Linux Kernel) je založena na jádru Linuxu verze 2.6. Platforma se kromě Linuxového jádra skládá i z C/C++ knihoven, např.: Grafické knihovny, SQLite, knihovny pro práci s médii, systémová knihovna libc, knihovny webového prohlížeče . . . [21]

Další součástí systému je Dalvik VM (Virtual Machine), který napsal Dan Bornstein. Dalvik je též jméno města na Islandu. Dalvik VM vykonává byte-code, kterým jsou reprezentovány vyšší vrstvy systému. Dalvik VM ale není Java VM a i jejich byte-cody se liší. Aby bylo možné přeložené byte-cody z Javy přenést na Android, obsahuje SDK nástroj *dx*, který provede převod Javového byte-codu (.class) na Dalvikův byte-code (.dex). Výsledný binární soubor je optimalizován pro běh na mobilních zařízeních.[**?**]aboutDalvik)

4.4 Výběr prostředí

Vývojáři Androidu doporučují Eclipse, do kterého je nutné přidat ADT plugin. Existují i pluginy pro NetBeans či Intellij IDEA. Rovněž je třeba nainstalovat SDK pro Android, bez kterého vývoj není možný. [18]

4.4.1 Android SDK

Standardní knihovna založená na jazyku Java. Android SDK obsahuje kromě knihoven i emulátor, který umožňuje ladění programů bez nutnosti jejich aplikace na fyzické zařízení. Emulátor je plně nastavitelný a lze pomocí něj vymodelovat vlastnosti konkrétního zařízení. V současné době ale nepodporuje Bluetooth. Dalším podstatným nástrojem je DDMS (Dalvik Debug Monitor Server), který slouží k ladění běhu aplikací. Zobrazuje komplexní informace o systému, jako jsou logy, běžící procesy, paměťové nároky procesů a také je schopen pořídit snímek obrazovky i u nerootových telefonů.[16, 3]

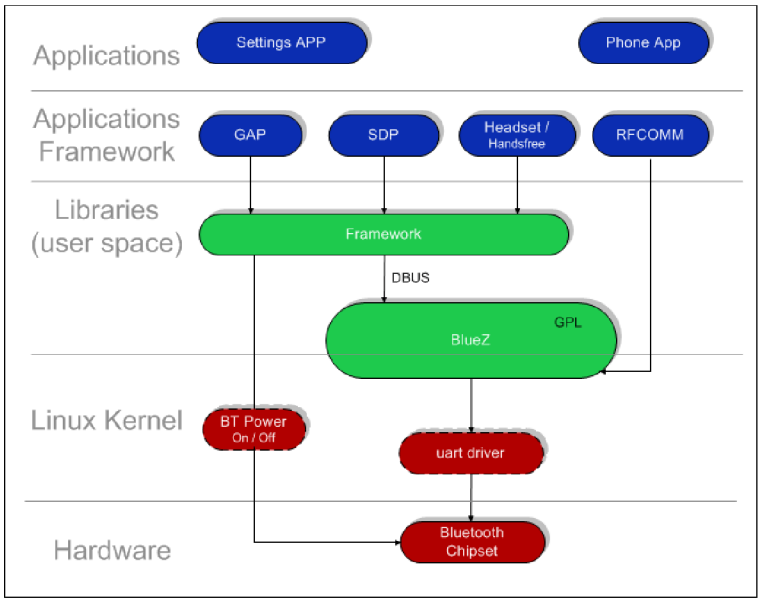
4.4.2 Android NDK

Pro aplikace, kde je třeba rychlá odezva, jako jsou hry anebo grafika, lze využít Android NDK, což je knihovna pro vývoj aplikací v C/C++. Typickým kanditátem pro použití NDK jsou aplikace, které intenzivně využívají CPU zařízení, ale nemají vysoké nároky na alokaci paměti. NDK je ve své podstatě náhradou a zmenšením existujícího jazyka C/C++. [17]

4.5 Bluetooth

Android stack na obrázku 4.3 je realizován pomocí knihovny BlueZ. BlueZ je šířena pod GPL licencí a Android Framework s ní komunikuje skrze D-BUS. Aby mohla aplikace pracovat s Bluetooth, musí mít nastavená příslušná oprávnění v konfiguračním souboru.

- *android.permission.BLUETOOTH* – povoluje komunikace (připojení k cíli, akceptace příchozího spojení, výměna dat)
- *android.permission.BLUETOOTH_ADMIN* – umožňuje manipulovat s nastavením (vyhledávat, být viditelný).



Obrázek 4.3: Diagram propojení knihoven Bluetooth. [15]

Kapitola 5

Specifikace aplikace

V této kapitole bude představena aplikace, kterou se zabývá tato práce a dále budou popsány funkce a vlastnosti aplikace, která je pojmenována jako Remote Controller.

Účelem aplikace je ovládat multimediální programy skrze Bluetooth. Aplikace je konstruována na principu komunikace klient - server. Server běží na počítači a po připojení klienta obsluhuje klientem zvolený program. Obsluha programu se provádí na základě skriptů, které si může tvořit sám uživatel. Při výběru aplikace je klientovi odeslán seznam podporovaných ovládacích a zobrazovacích prvků, které se vypíšou na displayi.

Program splňuje následující podmínky:

- Existuje skript pro ovládání aplikace a konfigurační soubor ovládacích prvků.
- Klient je schopen pracovat interaktivně (zobrazuje aktuální údaje o stavu přehrávače).
- Klient reaguje na chyby za běhu (nelze se připojit, spojení bylo ukončeno . . .).
- Klient zobrazuje právě ty ovládací prvky, které jsou uvedeny v konfiguračním souboru na serveru.
- Klient umožňuje posun času pomocí SeekBaru.
- Klient může vybírat stopy pro přehání z playlistu.
- Server běží jak na Linuxu, tak na Windows.
- Server posílá klientovi stavové informace o přehrávání.
- Párování zařízení neřeší aplikace, ale operační systém.
- Ovládání jendotlivých programů nedefinuje server, ale skripty.

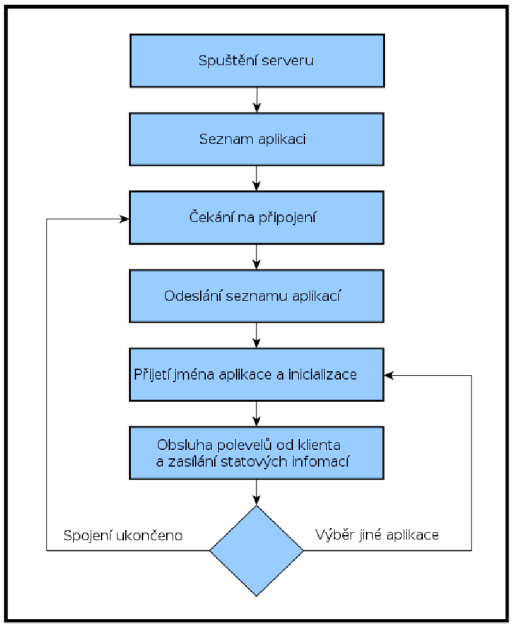
Kapitola 6

Návrh aplikace

V této kapitole se budeme zabývat popisem návrhu klienta i serveru.

6.1 Návrh serveru

Server je řízen na základě skriptů, které definují přehrávače a jejich chování. Server je použitelný jak pod Linuxem, tak i pod Windows. Vykonávání skriptů obstarává JavaScript, který je interpretován přímo v serveru. Při startu serveru je získán seznam dostupných skriptů a poté se čeká na připojení klienta. Po jeho připojení mu server odešle seznam ovladatelných aplikací. Při obdržení zprávy o výběru aplikace server provede inicializaci programu na základě informací ze skriptu. Poté server odešle klientovi seznam prvků, které lze dálkově ovládat. Při změně stavu ovládaného programu (změna jména souboru, posun času) server odešle změněné stavové informace klientovi. V případě chyby při zpracovávání skriptu reaguje server na tuto skutečnost adekvátním chybovým výstupem. Životní cyklus serveru je na obrázku 6.1.

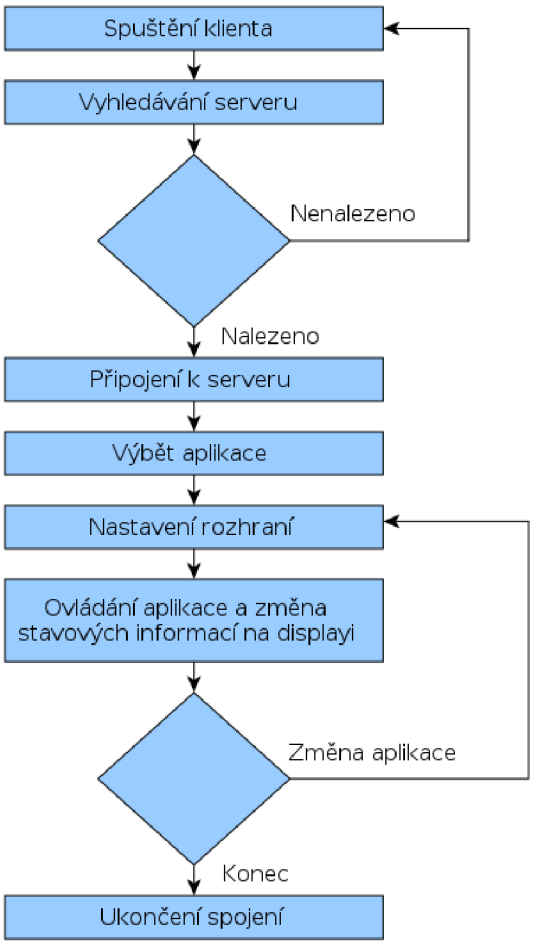


Obrázek 6.1: Životní cyklus server

6.2 Návrh klienta

Klient je napsán v Javě, za pomoci Android SDK. Díky této platformě by měl být schopen fungovat na všech zařízeních s operačním systémem Android, které obsahují Bluetooth. Do skupiny těchto zařízení nepatří jen mobilní telefony/smartphony, ale také poměrně rychle rozrůstající se kolekce tabletů.

Po spuštění klienta musí uživatel vybrat z hlavního menu položku *Připojit* a poté buď vybrat již spárované zařízení, nebo začít prohledávat okolí. Kliknutím na jméno zařízení se vyhledávání ukončí a aplikace se pokusí připojit k serveru. Pokud se spojení nezdaří, uživatel je informován na displayi. Po úspěšném připojení klient obdrží seznam možných ovladatelných aplikací, ze kterých si uživatel vybírá přes položku v hlavním menu *Vybrat aplikaci*. Jakmile klient odešle jméno aplikace, server mu odpoví seznamem možných ovladatelných prvků, které klient nastaví jako viditelné a nastaví obsluhu při kliknutí na obrázkové tlačítko, či změnu pozice v *SeekBaru*. Na displayi klienta se také pravidelně upravují informace o čase přehrávané skladby/videa a též název. Uživatel může kdykoliv za běhu změnit ovládanou aplikaci a je mu vygenerováno nové rozložení ovládání. Životní cyklus klienta je znázorněn na obrázku 6.2.



Obrázek 6.2: Životní cyklus klienta

6.3 Komunikace

Než budou zařízení moci mezi sebou komunikovat, musí se vytvořit spojení. Nejprve je třeba spustit server, který zaregistruje službu a čeká na příchozí spojení. Poté můžeme začít klientem prohledávat okolní zařízení a vybrat si, ke kterému se chceme připojit. Pokud připojení proběhlo úspěšně, může započít samotná komunikace.

Po připojení odesílá server klientovi zprávu, která obsahuje seznam aplikací, které jsou dostupné pro ovládání. Dále je odeslán i seznam příkazů pro ovládání počítače (např.:vypnutí počítače, restartování). Pak klient odešle jméno cílové aplikace, kterou chce ovládat. Server zašle klientovi seznam ovladatelných funkcí programu, provede jeho prvotní inicializaci a rozhodne, jakým způsobem program dále ovládat (zda je ovládán spuštěním jiného programu, nebo jestli příkazy zapisuje na standardní vstup běžícího procesu). Klient na základě informací o podporovaných ovládacích prvcích provede jejich zviditelnění na displayi. Poté již může klient ovládat program, nebo ukončit spojení. Pokud aplikace podporuje zasílání stavových informací, činí tak server pouze pro změněné údaje. Server po celou dobu spojení zasílá klientovi prázdné zprávy, které slouží jako detekce spojení. Server ztrátu spojení rozezná implicitně, klient bohužel vyžaduje tento druh kontroly. V případě ukončení nebo ztráty spojení se server restartuje a čeká na další spojení. Klient o stavu informuje uživatele a je připraven se opět připojit i k jinému stroji.

Kapitola 7

Implementace aplikace

V této kapitole bude popsána implementace klienta a serveru a také náhledy jejich grafických rozhraní. Pro implementaci klienta i serveru bylo použito prostření NetBeans [6]. Budou zde zmíněny i problémy spojené s implementací.

7.1 Implementace serveru

Zde bude popsána implementace serveru, jak vypadá jeho grafické rozhraní, a popis jednotlivých tříd. Bude také zmíněna spolupráce Javascriptu s ostatními aplikacemi a v neposlední řadě i problémy vzniklé při implementaci.

7.1.1 GUI

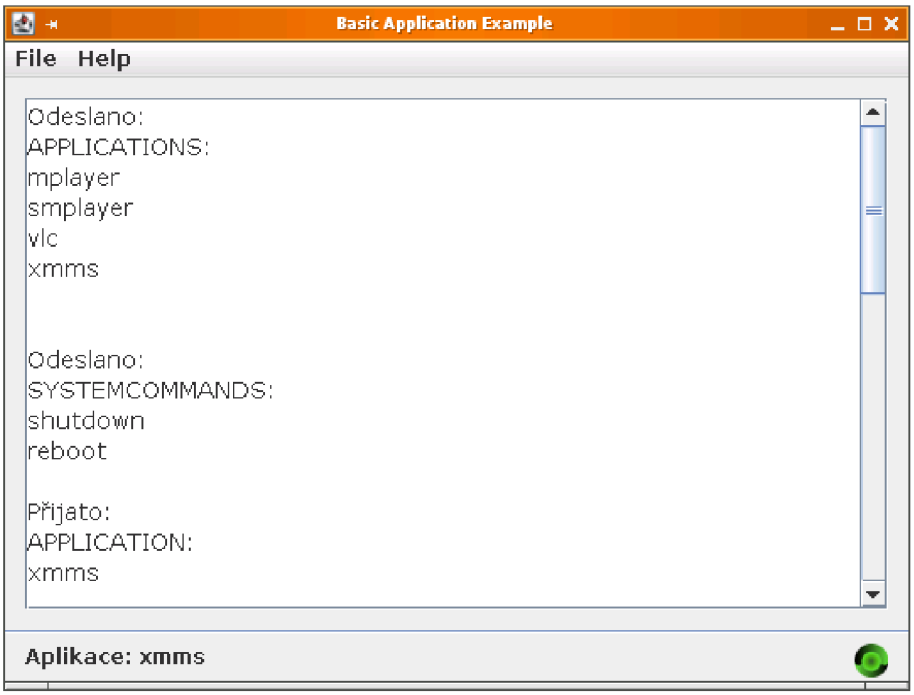
Grafické rozhraní je vytvořeno za pomoci balíku Java.swing, jež je novější variantou AWT. Swing umožňuje tvorbu dialogů, oken, tlačítek, seznamů ... AWT fungovalo jako rozhraní mezi Javou a API platformami. Všechny grafické komponenty byly kresleny přímo systémem. Na Windows se pro toto využívala grafická knihovna gdi32.dll. Tento přístup se nazývá jako těžký (heavy). Vzhled komponent tedy definuje systém, na kterém jsou zobrazovány.

Naproti tomu Swing pracuje na principu, kdy každá komponenta je zodpovědná za svůj vzhled a na požádání se vykreslí na monitor. Samotné kreslení provádí Java a systému je předán pouze obrázek toho, co se má zobrazit. Tento přístup se označuje jako lehký (lightweight). Swing navíc obsahuje i podporu pro dvojitou vyrovnávací paměť (double buffering), která umožňuje průhlednost nebo poloprůhlednost, či plynulé vykreslování grafických prvků.

Samotné okno je odvozené od *javax.swing.JFrame* a obsahuje prvek *javax.swing.jTextArea* pro zobrazení přijatých a odeslaných příkazů, který se před každým novým spojením vymaže. V dolní části okna je javax.swing.JPanel. Ten slouží jako stavový panel. Je na něm *javax.swing.JLabel*, který obsahuje jméno aktuálně ovládaného programu, a druhý *javax.swing.JLabel*, který slouží jako grafický indikátor, zda je někdo k serveru připojen či ne. Náhled grafického rozhraní serveru je na obrázku 7.1.

7.1.2 Třída AckTimer

Tato třída je potomkem třídy *Thread* a po připojení klienta mu v určitých intervalech zasílá zprávu *ACK*, aby byl klient schopen detekovat ztrátu spojení nebo jeho ukončení ze strany



Obrázek 7.1: Grafické rozhraní serveru

serveru. Jediným parametrem konstruktoru je odkaz na třídu, která obsahuje funkce pro odesílání zpráv skrze Bluetooth.

7.1.3 Třída BlueToothConnection

Je nutné, aby tato třída běžela jako vlákno, protože obstarává komunikaci s klientem a čeká na jeho připojení, a proto je potomkem třídy *Thread*. Po vytvoření instance se inicializuje zařízení a povolí se, aby byl server viditelný. Bluetooth server se spustí pomocí metody *run()*, která nastaví službu na základě URL a zavoláním metody *acceptAndOpen()* čeká na připojení klienta. Po připojení klienta server vytvoří instanci třídy *AckTimer* a zašle zprávu hlavní aplikaci o připojení klienta. Zasílání zpráv hlavní třídě *GUIView* probíhá pomocí *ActionListener*, *ActionListener* je do třídy předán metodou *addActionListener* a je využíván i pro zasílání upozornění, že byla přečtena nová zpráva od klienta nebo že sehlaho spojení. Získání obsahu přijaté zprávy probíhá zavoláním metody *getData()*.

7.1.4 Třída ApplicationsTerm

Třída funguje jako terminál pro vykonávání příkazů pro obsluhu programu. V konstruktoru je třídě předán flag o tom, jak se daný program ovládá (zda spuštěním nějakého programu s parametry, nebo se zapisuje na vstup běžícího programu). Pomocí funkce *executeCommand()* je předán příkaz, který se má spustit nebo zapsat na vstup běžícího procesu. Hlavní problém implementace této třídy je určit vhodnou časovou prodlevu mezi spuštěním procesu nebo zapsáním příkazu na jeho vstup a možností čtení relevantního výstupu. Je to řešeno ne zcela optimálně, a to několika uspáními a opakovaným testováním připravenosti pomocí metody *available()*. V případě pomalého počítače by tato metoda tento problém neodstranila. Třída se také snaží o zotavení běhu v případě, že uživatel ovládaný program

omylem ukončil.

7.1.5 Třída GUIView

Hlavní třída celé aplikace, jež vykresluje samotné grafické rozhraní a pomocí *ActionListener*u přijímá signály od ostatních tříd. Tím řídí chod celého programu. Při vytvoření objektu *BluetoothConnection* je vytvořen seznam ovládacích skriptů a konfiguračních souborů jednotlivých programů. Také probíhá zpracování případných možností pro ovládání počítače z klienta.

7.1.6 Třída OSDetection

Třída slouží pouze ke zjištění operačního systému, na kterém aplikace běží. Identifikuje tyto systémy:

- Unix/Linux
- Mac
- Windows

Typ operačního systému slouží pro řízení skriptu na ovládání počítače.

7.1.7 Třída ItemParser

Slouží k rozparsování konfiguračního souboru, který obsahuje informace o podporovaných ovládacích vlastnostech daného programu. V konstruktoru třídy je předán parametr jména konfiguračního souboru. Po zavolání metody *parse()* se provede samotné zpracování. V případě, že se během parsování vyskytla chyba, vratí metoda *false*. Dále třída obsahuje metodu *isStatusInfo()*, která vrátí *true* v případě, že program umožňuje zasílání stavových informací. Poslední metodou je *getMessage()*, která vrátí připravenou zprávu pro odeslání klientovi. V tabulce 7.1 jsou povolené položky souboru.

Jméno	**Popis**
prev	Přesun na předchozí stopu playlistu.
rew	Krátký posun zpět.
pause	Pozastavení přehrávání.
play	Přehrávání.
ff	Krátký posun vpřed.
playlist	Možnost zobrazení playlistu. Formát je *identifikátor stopy TAB Jméno*
seekBar	Povolení posuvníku při přehrávání.
statusInfo	Zasílání stavových informací

Tabulka 7.1: Povolené položky konfiguračního souboru

7.1.8 Třída JSParser

Jak již bylo řečeno v návrhu, ovládání programů probíhá pomocí skriptů v jazyce JavaScript, které převádějí přijaté příkazy od klienta na příkazy pro program. Skripty také zpracovávají

výstupy od aplikace, například přečtou a přetransformují playlist do požadovaného tvaru. Přímá podpora skriptování v Javě je od verze 6.[5]

Pro zprovoznění skriptového interpretu je třeba naimportovat *javax.script.ScriptEngine*, kvůli třídě *ScriptEngine*, která interpretuje skript. Pro zpřístupnění konkrétního skriptu použijeme třídu *ScriptEngineManager* a její metodu *getEngineByName()* s parametrem jména požadovaného skriptovacího jazyku. V našem případě tedy JavaScript. Pro zavolání některé funkce ze skriptu použijeme metodu *invokeFunction()* třídy *Invocable*.[5]

Každý skript musí obsashovat čtyři funkce:

- *initProgram()*
  Tato funkce vrací příkaz, který je nutné provést před samotným ovládáním programu.
- *isStreamController*
  Funkce vrací *true* v případě, že je program ovládán zapisováním příkazů na standardní vstup. *False* vrací pokud je program ovládán spouštěním programu s parametry.
- *getCommand(action, param)*
  Funkce překládá hodnotu proměnné *action* na příkaz pro řízení programu. Druhý parametr *param* je v určitých situacích vyžadován pro předání hodnoty do programu (např.: *trace +cas*, kde *trace* je v proměnné *action* a *+cas* v proměnné *param*).
- *getResult(action, result)*
  Tato funkce má za úkol zpracovávat výstupní informace z programu a přepracovat je do podoby potřebné pro transport ke klientovi (např.: převod playlistu do potřebného formátu). Proměnná *action* obsahuje jméno povelu pro program a proměnná *result* výstup programu.

V případě chyby ve skriptu je o tom informován uživatel jak na straně klienta, tak v textové oblasti určené pro výpisy na serveru.

7.1.9 Třída MFiles

Slouží k uchovávání informací o ovladatelných aplikacích. Obsahuje cestu ke skriptu, cestu ke konfiguračnímu souboru a samotné jméno aplikace.

7.1.10 Třída MessageParser

Tato třída slouží k rozparsování přijaté zprávy od klienta. Jediným parametrem konstruktoru je zpráva. Po zavolání metody *parse()* se provede její rozparsování. Konec zprávy musí být zakončen prázdným řádkem. Zprávy jsou ve tvaru např.:

```
APPLICATION:
xmms
```

Tato zpráva znamená, že uživatel chce ovládat program *xmms*. V tabulce 7.2 jsou uvedené implementované zprávy.

7.1.11 Třída SendStatusThread

Jak napovídá název, třída je opět vláknem. Slouží pro pravidelné posílání stavových informací z ovládaných programů (jméno stopy, celkový čas, aktuální čas). Třída je použita pouze v případě, že ovládaný program umožňuje získávání těchto informací.

Jméno	Popis
APPLICATION	Uživatel si vybral aplikaci, kterou chce ovládat
COMMAND	Příkaz, který se má provést
REMOTECOMMAND	Příkaz na ovládání systému

Tabulka 7.2: Přijímané zprávy od klienta

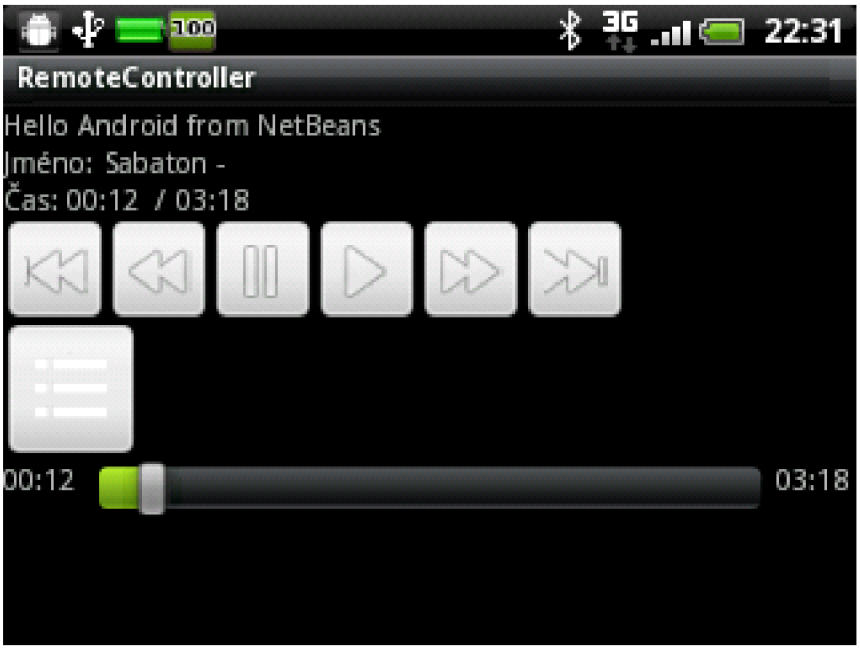
7.2 Implementace klienta

7.2.1 GUI

Pro tvorbu grafického rozhraní pro Android je zapotřebí importovat balík *android.app.Activity*, který je vlastně plochou, na niž se skládají jednolivé komponenty. Předpis grafického rozhraní je uložen v XML souboru *main.xml*. Vzhled každé další *Activity* je definován dalším XML souborem. Pomocí nich lze upravovat vzhled i jednotlivých komponent. Pro tvorbu rozhraní byly použity tyto komponenty:

- *TextView* – Textové pole pro umístnění popisků.
- *ImageButton* – Je použito pro grafické ztvárnění jendotlivých ovládacích tlačítkových prvků.
- *SeekBar* – Posuvník pro přeskakování v rámci stopy.

Při startu programu jsou všechny komponenty skryty a jsou zobrazeny až po přijetí rozložení ovládacích prvků od serveru. Na obrázku 7.2 je náhled na grafické rozhraní.



Obrázek 7.2: Grafické rozhraní klienta

Položky menu (jiné obrazovky) jako je vyhledávání serveru, ovládání počítače, výběr aplikace a i playlist jsou také realizovány pomocí *android.app.Activity*.

7.2.2 Třída ButtonsController

Tato třída se stará o inicializaci ovládacích prvků aplikace. Obsahuje metodu *setInvisible()*, která skryje všechny prvky a nastaví jim výchozí hodnoty (např.: časy nastaví do výchozí hodnoty - 00:00). Dále je zde metoda *setVIsible()*, která jako parametr dostane přijatý seznam ovládacích prvků o serveru a na jeho základě zviditelní dané prvky a nastaví jim akci při stisknutí. Další důležitou metodou je *setConnectin()*. Ta předá do objektu objekt *ConnectThread*, pomocí kterého inicializuje zápis povelů do Bluetooth při stisku tlačítka.

7.2.3 Třída ConnectThread

Třída je potomkem třídy *Thread* a má za úkol vytvořit spojení se severem. Jako parametr konstruktoru dostává *BluetoothDevice*, pro vytvoření spojení, a *Handler*, pro komunikaci s hlavní třídou.

7.2.4 Třída ConnectedThread

Je vlákno, ve kterém běží nekonečná smyčka čtení ze socketu. V případě přijetí zprávy se zkontroluje, zda končí na
n
n. Pokud ne, čeká se na pokračování zprávy, které toto ukončení obsahuje. Poté je zpráva pomocí třídy *Handler* poslána do hlavního programu. Ve smyčce vlákna se také kontroluje, kdy naposledy přišel signál od serveru. V případě, že je toto prodlení příliš dlouhé, odesílá se chyba do hlavního programu.

7.2.5 Třída ListDeviceActivity

Tato třída je potomkem třídy *Activity* a slouží pro vykreslení obrazovky, která obsahuje seznam již spárovaných zařízení a ve které se zobrazují průběžně nalezená další zařízení. Pro výpis spárovaných a nalezených zařízení se používá komponenta *android.widget.ListView*. Tlačítko hledat je komponenta *android.widget.Button*. V konstruktoru třídy se také provede registrace příjemce zpráv o nalzení zařízení v okolí. Po stisknutí tlačítka Hledat se zobrazí *ProgressBar* aktivity. V případě nálezu zařízení je jeho jméno a MAC adresa a umístěna do oblasti *Nalezená zařízení*. Jakmile příjmce zpráv obdrží *ACTION_DISCOVERY_FINISHED*, skryje se *ProgressBar* a v případě nenalezení žádného zařízení se vypíše toto sdělení na obrazovku. V průběhu celého procesu vyhledávání může uživatel kdykoliv stiknout jméno nalezeného, nebo již spárovaného zařízení pro ukončení vyhledávání a předání výsledku do hlavní aplikace.

7.2.6 Třídy ListRemoteApplication, ListPlayList, ListRemoteSystem

Třídy jsou postaveny na stejném principu seznamů jako třída *ListDeviceActivity*. Třídy mají tuto funkci:

- *ListRemoteApplication*
  Slouží pro výběr dálkově ovládané aplikace.
- *ListPlayList*
  Určená pro výběr stopy z playlistu zaslaného serverem.

- *ListRemoteSystem*
  Třída pro výběr možnosti ovládání vzdáleného systému (vypnout, restartovat . . .).

7.2.7 Třída RemoteController

Třída hlavní aplikace, která je potomkem *android.app.Activity* a na jejíž ploše jsou zobrazeny ovládací prvky. S touto třídou je spjato i hlavní menu, ze kterého se přistupuje k připojení k serveru, výběru aplikace a k ovládání vzdáleného počítače. Třída obsahuje metodu *onActivityResult()*, která přijímá výsledky od jiných zavolaných tříd odvozených od *android.app.Activity*. Obsluhované kódy jsou uvedeny v tabulce 7.4

K´od	**Popis**
REQUEST_CONNECT_BLUETOOTH	Požadavek na připojení k serveru. Jakmile si uživatel vybere cílové zařízeni z nabídky *ListDeviceActivity*, je tato akce odchycena zde a provede pokus o spojení.
REQUEST_SELECT_APPLICATION	V případě, že si klient vybere z *ListRemoteApplication* aplikaci, je zpráva o tom zachycena zde a posléze je odesláno jméno aplikace na server.
REQUEST_SELECT_REMOTE_COMMAND	Klient si vybral z *ListRemoteApplication* příkaz pro server a jméno příkazu je odesláno.
REQUEST_SELECT_TRACK	Požadavek na výběr stopy z playlistu (*ListPlayList*). Jméno stopy je odesláno serveru.

Tabulka 7.3: Kódy obsluhované v metodě *onActivityResult()*

Další podstatnou metodou třídy je *handlerMessage()*, která přijímá zprávy od ostatních tříd a podle jejich obsahu řídí další běh aplikace. Seznam zprávy je v tabulce 7.4.

Zpráva	**Popis**
STATE_CONNECTED	Klient byl připojen k severu.
MESSAGE_READ	Byla přijata zpráva od serveru. Zpráva je zde rozparsována a zpracována.
ERROR_CONNECTION	Nezdařilo se připojení k severu. Klient se uvede do stavu jako při spuštění.
ERROR_CONNECTION_LOST	Spojení bylo ztraceno. Stejný postup jako u ERROR_CONNECTION.

Tabulka 7.4: Zprávy pro řízení běhu klienta.

7.3 Třída RemoteParser

Třída slouží k rozparsování zprávy od severu. Typy zpráv jsou uvedeny u implementace serveru.

Kapitola 8

Závěr

Cílem balakářské práce bylo vytvořit aplikaci pro mobilní telefony s operačním systémem Android, která umožňuje ovládat audio a video programy na počítači. Cílovou platformou měl být Linux, ale výsledná aplikace počáteční požadavky předčila a je použitelná i na Windows. Implementačním jazykem byla na straně počítače Java a u mobilního telefonu Java s využitím Android SDK.

Aplikace byla napsána co možná neuniverzálněji. Uživatel je schopen si sám pomocí JavaScriptu přídávat další a další aplikace, které může pomocí telefonu ovládat.

Literatura

[1] BlueCove - BlueCove JSR-82 project [online]. http://bluecove.org/index.html, 12-25-2008 [cit. 2011-04-20].

[2] Bluetooth [online]. http://en.wikipedia.org/wiki/Bluetooth, 15 May 2001, rev. 30 November 2010 [cit. 2010-11-30].

[3] Vyvíjíme pro android – úvod [online]. http://www.abclinuxu.cz/clanky/vyvijime-pro-android-uvod, 16.3. 2011 [cit. 2011-04-1].

[4] Java SE 6 Documentation [online]. http://download.oracle.com/javase/6/docs/, 1993, rev. 2011 [cit. 2011-04-20].

[5] Java Scripting Programmer's Guide [online]. http://download.oracle.com/javase/6/docs/technotes/guides/scripting/programmer_guid 1993, rev. 2011 [cit. 2011-04-22].

[6] Welcome to NetBeans [online]. http://netbeans.org, 2011 [cit. 2011-04-22].

[7] Java (programovací jazyk) [online]. http://cs.wikipedia.org/wiki/Java_(programovací_jazyk), 22. 6. 2004, rev. 13. 3. 2011 [cit. 2011-04-20].

[8] Bluetooth profile [online]. http://en.wikipedia.org/wiki/Bluetooth_profile, 22 September 2006, rev. 3 January 2011 [cit. 2011-1-10].

[9] Bluetooth [online]. http://cs.wikipedia.org/wiki/Bluetooth, 25. 6. 2005, rev. 5. 11. 2010 [cit. 2010-11-30].

[10] Personal area network [online]. http://en.wikipedia.org/wiki/Personal_area_network, 25 February 2002, rev. 12 April 2011 [cit. 2011-04-20].

[11] Android (operation system) [online]. http://en.wikipedia.org/wiki/Android_(operating_system), 4 August 2007, rev. 9 January 2011 [cit. 2011-1-10].

[12] Java Platform, Standard Edition [online]. http://en.wikipedia.org/wiki/Java_Platform,_Standard_Edition, 4 March 2002, rev. 15 April 2011 [cit. 2011-04-20].

[13] IrDA [online]. http://cs.wikipedia.org/wiki/IrDA, 8. 12. 2005, rev. 16. 1. 2011 [cit. 2011-04-20].

[14] Wi – fi [online]. http://cs.wikipedia.org/wiki/Wi-Fi, 9. 11. 2004, rev. 13. 4. 2011 [cit. 2011-04-20].

[15] Bluetooth [online]. http://wing-linux.sourceforge.net/online-pdk/guide/bluetooth.html, [cit. 2010-11-30].

[16] Installing the SDK — Android Developers [online]. http://developer.android.com/sdk/installing.html, [cit. 2010-11-30].

[17] What is NDK [online]. http://developer.android.com/sdk/ndk/overview.html, [cit. 2010-11-30].

[18] ADT Plugin for Eclipse [online]. http://developer.android.com/sdk/eclipse-adt.html, [cit. 2011-1-15].

[19] Secure your Bluetooth wireless networks [online]. http://www.zdnetasia.com/i/news/2006/olzak_bluetooth_b.jpg, December 4, 2006 [cit. 2011-04-20].

[20] J2ME APIs: Which APIs come from the J2SE Platform? [online]. http:// developers.sun.com/ mobility/midp/articles/api/, January 2001 [cit. 2011-04-20].

[21] What is Android? — Android Developers [online]. http://developer.android.com/guide/basics/what-is-android.html, rev. 11 Nov 2010 [cit. 2010-11-30].

[22] Herout, P.: *Učebnice jazyka Java*. České Budějovice: KOPP, 2003, ISBN 80-7232-116-3.

[23] Valenta, T.: *Využití Bluetooth přenosu v J2ME aplikacích - BT MEssenger*. Bakalářská práce, ČVUT v Praze – FEL, Praha, 2006.

[24] Čepička, D.: Základy technologie Bluetooth: původ a rozsah funkcí [online]. http://pcworld.cz/hardware/Zaklady-technologie-Bluetooth-puvod-a-rozsah-funkci-6635 10.02.2009 [cit. 2010-12-5].

Seznam obrázků

Seznam tabulek