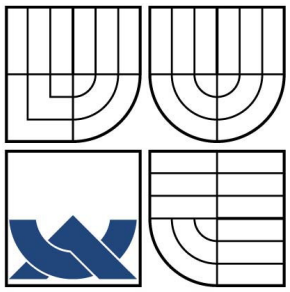
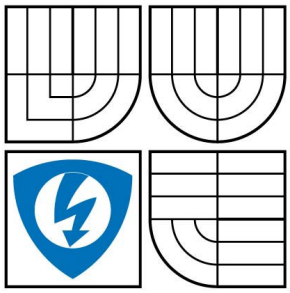


VYSOKÉ UČENÍ TECHNICKÉ
V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

Video on Demand v JavaME

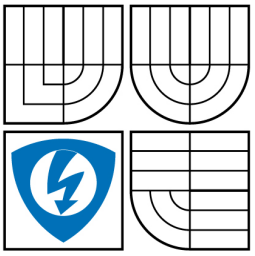
Video on Demand in JavaME

SEMESTRÁLNÍ PRÁCE
SEMESTRAL THESIS

AUTOR PRÁCE — Bc. Petr OBDRŽÁLEK
AUTHOR

VEDOUCÍ PRÁCE — Ing. Petr ČÍKA
SUPERVISOR

BRNO 2008



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Diplomová práce

magisterský navazující studijní obor
Telekomunikační a informační technika

Student:	Bc. Petr Obdržálek	***ID:***	80514
Ročník:	2	***Akademický rok:***	2008/2009

NÁZEV TÉMATU:

Video na vyžádání v JavaME

POKYNY PRO VYPRACOVÁNÍ:

Prostudujte možnosti přenosu videa do mobilních telefonů v reálném čase. Navrhněte aplikaci umožňující provozovat službu video a audio na vyžádání na mobilním telefonu. K přenosu využijte protokoly RTP a RTCP, k ovládání médií protokol RTSP. Navrženou aplikaci realizujte v prostředí JavaME. Zároveň vytvořte databázi audio a video nahrávek, která bude dostupná uživatelům po jejich registraci a uhrazení polatku za službu. Pro vytvoření databáze využijte jazyk SQL, pro její řízení PHP.

DOPORUČENÁ LITERATURA:

[1] Yuan, J., Y.: Enterprise J2ME: Developing Mobile Java Applications, Indiana, Prentice Hall PTR 2003, ISBN 978-0131405301
[2] Wells, M., J.: Java ME Game Programming, 2E, Florence, Course Technology PTR 2007, ISBN 978-1598633894

Termín zadání:	9.2.2009	***Termín odevzdání:***	26.5.2009

Vedoucí práce: Ing. Petr Číka

prof. Ing. Kamil Vrba, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

Anotace

Diplomová práce je zaměřena na vytvoření systému pro přehrávání videa na vyžádání. Jsou rozebírány technologie, které se používají při tvorbě aplikací pro mobilní zařízení. Také jsou uvedeny dnes nejpoužívanější kodeky v mobilních zařízeních. Jsou zmíněny standardy, normy, principy a doporučení pro přenos multimédií přes síť v reálném čase. Dále jsou popsány technologie, které je vhodné použít na serverové straně pro funkčnost navrženého systému. Závěrem práce je vypracování funkčního vzorku celého systému a popis funkčnosti tohoto systému.

Klíčová slova

Java ME, video na vyžádání, RTP, RTCP, RTSP, PHP, MySQL, XHTML, CSS

Abstract

The master’s thesis deals with creation of system that provides video on demand. Technologies which are used to creation mobile application are analyzed. There are also mentioned today's most used codecs in the mobile devices. There are described standards, norms, principles and recommendations for transfer multimedia data on network in real time. Technologies which are appropriate for functionality of system on server side are described. The output of the work is an operational sample of whole system and description of functionality of this system.

Keywords

Java ME, video on demand, RTP, RTCP, RTSP, PHP, MySQL, XHTML, CSS

Citace práce

OBDRŽÁLEK, P. *Video na vyžádání v JavaME.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2009. 55 s. Vedoucí diplomové práce Ing. Petr Číka.

Prohlášení

Prohlašuji, že svoji diplomovou práci na téma Video on Demand v JavaME, jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením tohoto projektu jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

V Brně dne …… ..…….

podpis autora

Poděkování

Děkuji vedoucímu diplomové práce Ing. Petru Číkovi, za velmi užitečnou metodickou pomoc, cenné rady a praktické ukázky při zpracování diplomové práce.

V Brně dne …… ……..

podpis autora

Obsah

Seznam obrázků

Úvod

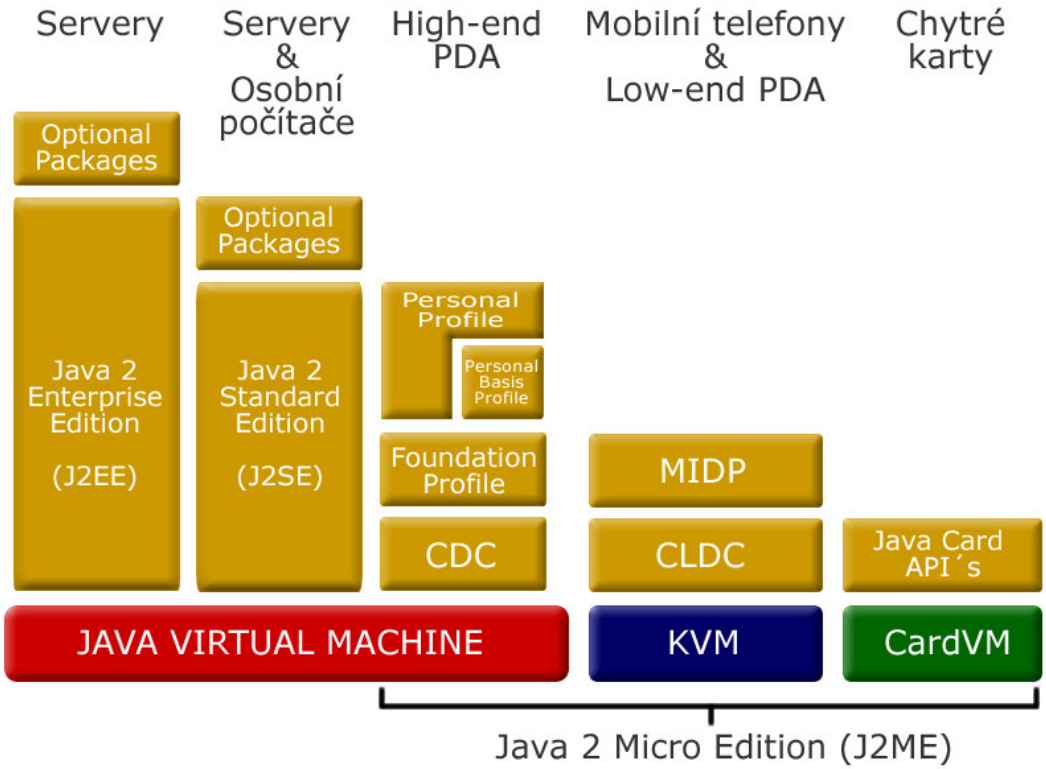
Mobilní zařízení jsou dnes nepostradatelnou součástí běžného života. V počátcích svého vzniku sloužily hlavně pro komunikaci mezi lidmi. Postupem času se však začaly stávat víceúčelovými zařízeními, kde komunikace je jedna z mnoha jejich funkcí. Mezi nejvíce využívané funkce dnešních mobilních telefonů patří bezesporu multimédia. Mnoho mobilních telefonů je přímo stavěno na práci s multimédii a nahrazují tak osobní multimediální přehrávače.

Dnešní mobilní zařízení existují v mnoha variacích, především podle jejich primárního zaměření. Jmenujme alespoň ty nejhlavnější, jako jsou manažerské telefony, multimediální telefony, designérské skvosty a další. Mobilní telefony se ale hlavně liší svým výpočetním výkonem a dostupnou operační pamětí. Výrobci nabízejí programátorům aplikací pro tyto terminály množství API, které však většinou nejsou plně kompatibilní. Takto vytvořené programy mají pak malou škálu použitelnosti. JavaME nabízí řešení pro tento problém. Zavádí jednotné rozhraní pro širokou škálu mobilních zařízení. Dále jsou diskutovány potřebné technologie pro zprovoznění navrženého systému, který bude schopen přenosu videa na vyžádání.

Tato práce se zabývá problematikou multimédií v mobilních telefonech při použití platformy JavaME. Je popsána historie a struktura této platformy. Praktická část popisuje vytvořený přehrávač pro přehrávání multimediálních souborů uložených na lokálním úložišti a také přehrávání streamovaného obsahu přes síť. Aplikace nabízí základní funkce pro přehrávání multimédií. Druhou částí bude vytvoření serverového řešení celého systému, aby korektně fungoval přenos videa na vyžádání.

1. Jazyk JAVA

Java je objektově orientovaný programovací jazyk vyvinutý firmou Sun Microsystems. Jedná se o jeden z nejpoužívanějších programovacích jazyků na světě. Javu lze díky její přenositelnosti použít téměř všude. Počínaje čipovými kartami (smart cards – platforma JavaCard), přes mobilní telefony, set-top boxy a různá zabudovaná a přenosná zařízení (platforma JavaME), aplikace pro osobní počítače (platforma JavaSE), až po rozsáhlé distribuované systémy pracující na serverech po celém světě (platforma JavaEE).



Obr. 1: Technologie JAVA

1.1. Historie jazyka JAVA

Java původem vychází z programovacího jazyka Oak vyvíjeného společností Sun Microsystems určeného pro programování malých spotřebních zařízení. Tento jazyk byl navržen pro tvorbu spolehlivého softwaru, který by byl nenáročný na paměť a výkon zařízení. Nebyl však nikdy k těmto účelům využit. Při rozmachu internetu společnost Sun Microsystems přejmenovala jazyk Oak na Java. Pomocí Javy pak vytvořila internetový prohlížeč HotJava, který byl přenositelný. Následovalo začlenění Javy do populárního prohlížeče Netscape v podobě Java appletů, čímž se dostala do povědomí veřejnosti.

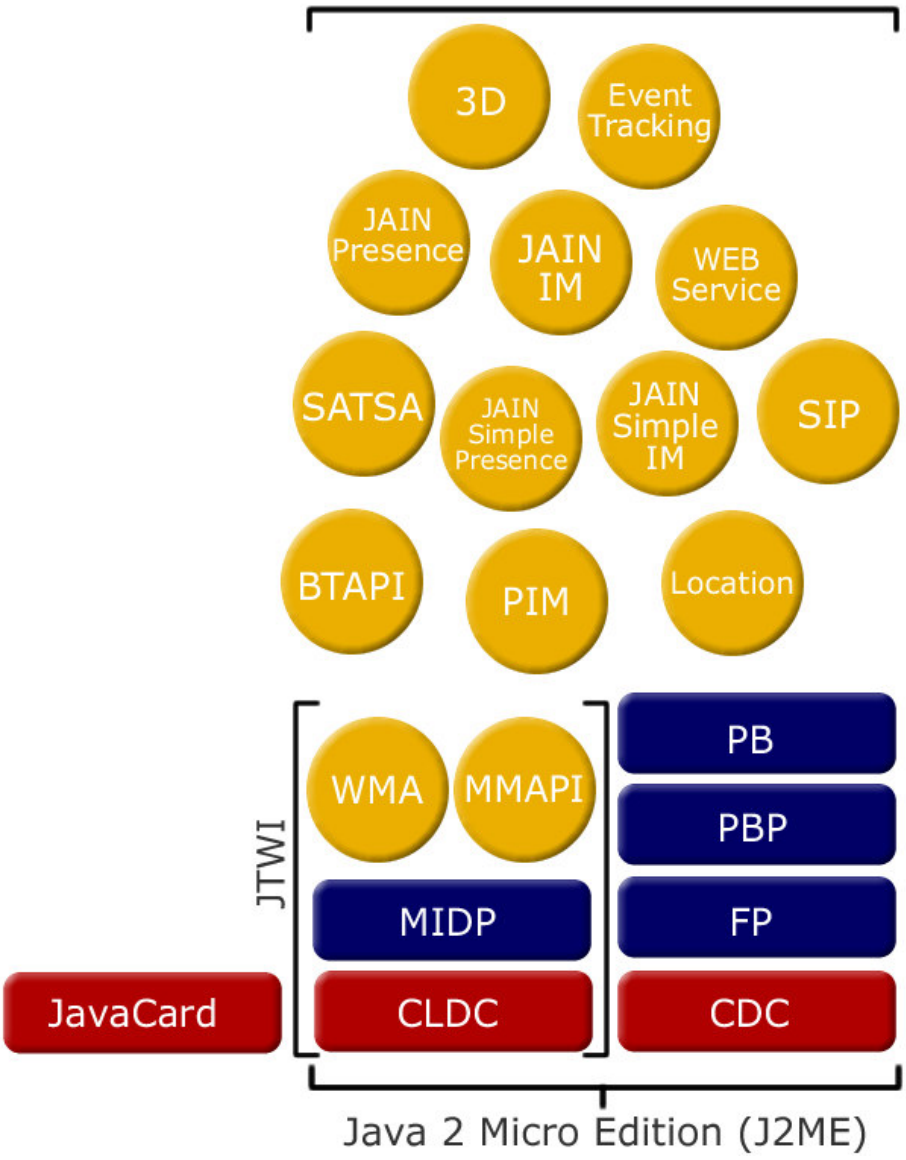
Během několika let se stává Java stále oblíbenější především díky své přenositelnosti. Před vydáním Javy 2 se platforma rozděluje na J2SE (Java 2 Standard Edition) a J2EE (Java 2 Enterprise Edition).

Spolu s rozvojem mobilních technologií a zařízení roste poptávka po platformě, která bude vhodná pro vývoj aplikací pro malá zařízení. Stávající platforma J2SE není vhodná kvůli velkým nárokům na paměť a výpočetní výkon zařízení. Sun tedy začíná vyvíjet několik platforem založených na JDK 1.1, přičemž každá platforma je vhodná k jiným účelům. JavaME je pak sjednocením těchto platforem na úrovni Java 2.

2. JavaME

2.1. Architektura

JavaME je prostředí pro programy napsané v jazyce Java určené k vytvoření programů od nejjednodušších čipových karet až po výkonná zařízení. Používá se v mobilních telefonech, pagerech, navigačních systémech, set-top boxech a dalších zařízeních. Výhodou JavaME je to, že jeden program může teoreticky fungovat na několika různých zařízeních, neboť jazyk je pořád stejný a to Java. Vzhledem k této rozmanitosti ale není dostatečně možné přistupovat ke všem zařízením stejně. Proto existuje několik konfigurací, profilů a volitelných balíčků, pomocí nichž lze získat kompletní API pro vývoj na daném zařízení.



Obr. 2: Schéma platformy JavaME

2.1.1. Konfigurace

Konfigurací rozumíme množinu základních tříd (core classes) a parametry JVM (Java Virtual Machina). Zařízení nemusí obsahovat celou JVM, ale pouze její podmnožinu (KVM, CVM). Výběr konfigurace závisí na daném zařízení pro které je aplikace vyvíjena a na jeho základních charakteristikách, mezi něž patří například frekvence procesoru nebo dostupná paměť. V současné době jsou definovány dvě konfigurace CLDC a CDC, v budoucnu se předpokládá vznik další konfigurace v závislosti na vývoji zařízení.

Conected Limited Device Configuration (CLDC) je určena pro malá zařízení s omezenými zdroji. Tato konfigurace a KVM (Kilobyte Virtual Machina) se používají pro malé aplikace. Programátor musí respektovat výkonové a paměťové omezení jednotlivých přístrojů. Existuje ve verzi CLDC 1.0 definované specifikací JSR-30 a CLDC 1.1 podle specifikace JSR-139.

Cíle konfigurace CLDC:

- Nízké nároky na zdroje
- Zaměření na aplikační programy ne na systémové
- Podpořit vývoj aplikací třetích stran

Druhou konfigurací je **Conected Device Configuration** (CDC), která je určena pro výkonnější zařízení s větší pamětí. Vyplňuje díru mezi CLDC a JavaSE. Jako virtuální stroj je použito CVM. CDC existuje ve verzích CDC 1.0 (JSR-36) a CDC 1.1 (JSR-218).

Cíle konfigurace CDC:

- Podpora vzniku sofistikovanějších aplikací – díky většímu displeji, výkonu, paměti a dostupným knihovnám je možné vytvořit i složitou aplikaci

2.1.2. Profily

Profil je doplněk ke konkrétní konfiguraci a dodává jí další funkcionalitu pro určitou množinu zařízení. Je postaven nad konfigurací. Jednotlivé profily mohou být mezi sebou provázány.

Profily konfigurace CLDC:

Mobile Information Device Profil (MIDP)

MIDP je nejznámější součástí JavaME platformy a nejvíce používaný profil nad konfigurací CLDC. Tvoří základ technologie Wireless Java. Aplikace napsané pod

MIDP se nazývají MIDlety. MIDP existuje v několika specifikacích. MIDP 1.0 (JSR-37) poskytuje API potřebná k vytvoření uživatelského rozhraní (UI) a síťové služby. MIDP 2.0 (JSR-118) rozšiřuje MIDP 1.0 o API na přehrávání multimédií, herní API zaměřené na 2D hry a autentizační API pro zabezpečené připojení. Nejnovější specifikace MIDP 3.0 (JSR-271) je stále ve vývoji.

Information Module Profile (IMP)

IMP vychází z profilu MIDP a nabízí prostředí pro zařízení s omezenými zobrazovacími schopnostmi. Nabízí například API pro síťové sužby nebo datová úložiště. Používá se třeba v parkovacích automatech nebo v bezdrátových modulech domácích bezpečnostních zařízeních. IMP je zatím pouze ve verzi 1.0 (JSR-195). Probíhá však vývoj nové verze IMP Next Generation (JSR-228).

Profily konfigurace CDC:

Foundation Profile (FP)

Určen pro zařízení, která potřebují částečnou nebo úplnou podporu JavySE. Neobsahuje žádné uživatelské rozhraní (GUI). Z tohoto profilu pak vychází i ostatní.

Personal Basis Profile (PBP)

Obsahuje totéž co Foundation Profile, ale navíc přidává část knihovny AWT (Abstrakt Windows Toolkit) což je základní uživatelské rozhraní, které je však omezeno na použití pouze jednoho okna.

Personal Profile (PP)

Navíc obsahuje náročnější komponenty AWT a vylepšenou JVM. Je rozšířena na spoustě zařízení, jako jsou PDA nebo komunikátory (Nokia 9210).

2.1.3. Volitelné balíčky

Volitelné balíčky rozšiřují platformu JavaME a přidávají další funkcionality ke konfiguracím a profilům, která není buď dostatečně obecná, aby mohla být začleněna do profilu, nebo je sdílena různými profily. Rozšiřují například oblast práce s databázemi, wireless messaging, multimédii, 3D grafikou a webovými službami. Každý výrobce pak implementuje podporu jen těch balíčků, které jsou vhodné pro jeho zařízení. [1], [5]

2.2. *Struktura programu v JavaME*

V této kapitole bude popsán běh MIDletu, jeho kostra, a třídy potřebné pro vývoj aplikace.

Konfigurace CLDC obsahuje tyto třídy:

- Java.io - vstup a výstup pomocí datových proudů
- Java.lang - třídy jazyka java (Integer apod.)
- Java.util - kolekce a funkce pro čas a datum
- Javax.microedition.io – obecné funkce pro konektivitu

Jde vidět, že pro vývoj použitelných programů je potřeba tyto třídy rozšířit o další. Hlavně z důvodu komunikace s uživatelem. Tuto možnost nám poskytuje profil MIDP.

Balíčky tříd MIDP 2.0:

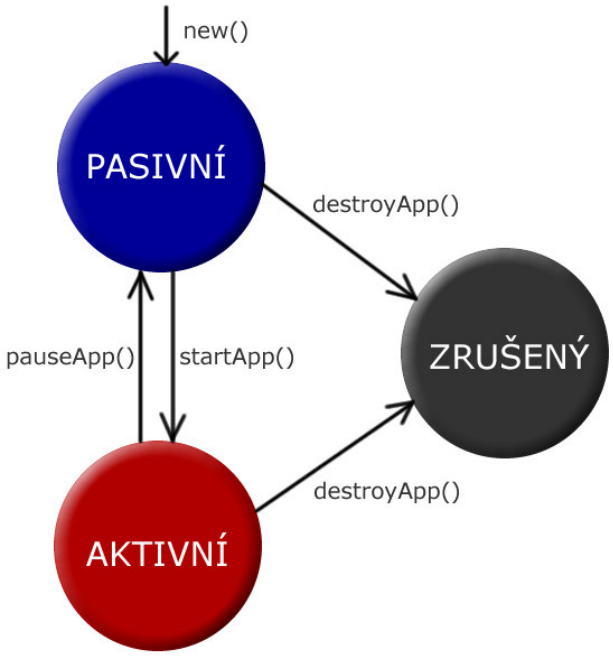
- Javax.microedition.lcdui - uživatelský interface
- Javax.microedition.rms - persistence dat –
- Javax.microedition.midlet - životní cyklus aplikace
- Javax.microedition.io - síťové funkce
- Javax.microedition.media - audio
- Javax.microedition.pki - autentizace

Základní programovou jednotkou je tzv. MIDlet. MIDlet je aplikace v Javě. Některými vlastnostmi se podobá appletu. Musí rozšiřovat speciální třídu MIDlet a běží z bezpečnostních důvodů v tzv. sandboxu, neboli na vlastním pískovišti, které nemůže opustit. Pomocí MIDletu jsme schopni řídit životní cyklus aplikace. Aplikaci vytváříme tak, že nejprve vytvoříme potomka třídy MIDlet a předefinujeme některé jeho metody. Prázdný MIDlet obsahuje tři základní metody `startApp()`, `pauseApp()` a `destroyApp()` (Obr. 3):

```
import javax.microedition.midlet.*;
public class Main extends MIDlet {
    public void startApp() {
    }
    public void pauseApp() {
    }
    public void destroyApp(boolean unconditional) {
    }
}
```

Obr. 3: Kostra MIDletu

MIDlet se může nacházet v několika stavech, přičemž programátor může na každý stav reagovat. MIDlet mezi těmito stavy přechází buď následkem vnější události (uživatel přerušil běh aplikace apod.), nebo na žádost programu. Životní cyklus MIDletu je vidět na Obr. 4. Běh aplikace a její přechody mezi stavy řídí aplikační manažer. [1]



Obr. 4: Životní cyklus MIDletu

Stavy v životním cyklu midletu:

- **Pasivní** – MIDlet je inicializován, ale neběží. V tomto stavu by neměl vlastnit ani využívat žádné sdílené zdroje.
- **Aktivní** – MIDlet je v normálním režimu a využívá sdílených zdrojů.
- **Zrušený** – MIDlet uvolní využívané prostředky a neběží. Do tohoto stavu se může dostat jen jednou.

pauseApp	Při tomto přechodu by měl MIDlet uvolnit všechny prostředky
startApp	MIDlet si alokuje zdroje a pokračuje v běhu
destroyApp	MIDlet uloží svůj stav a uvolní alokované zdroje
notifyDestroyed	Metoda pro upozornění manažeru aplikací, že MIDlet uvolnil zdroje a je připraven k ukončení
notifyPaused	V tomto stavu upozorňuje manažera, že přešel do stavu Pause (Pasivní)
resumeRequest	MIDlet žádá touto metodou řídící software o opětovné spuštění
getAppProperty	MIDlet se dotazuje na vlastnosti běhového prostředí

Tabulka 1: MIDlet interface

2.3. Zabalení aplikace

Aplikace je složena ze dvou souborů a to JAD a JAR. Jejich struktura je daná specifikací profilu MIDP. Tato dvojice má název sada MIDletů (midlet suite).

V souboru JAR je zabalen jeden nebo více MIDletů. Tento soubor může také obsahovat:

- manifest soubor,
- Java třídy pro MIDlet,
- další zdroje jako například obrázky, zvuky a další.

Manifest soubor se nachází v JAR souboru v adresáři META-INF s názvem manifest.mf. Musí obsahovat všechny povinné atributy jinak ho některé telefony můžou odmítnout nainstalovat z důvodu bezpečnosti.

Název atributu	Povinný	Popis
MIDlet-Name	Ano	Název sady MIDletů, kterým se identifikuje uživateli
MIDlet-Version	Ano	Číslo verze sady MIDletů používané manažerským softwarem v telefonu při aktualizaci
MIDlet-Vendor	Ano	Výrobce sady MIDletů
MIDlet-Icon	Ne	Jméno png souboru uvnitř JAR souboru. Tento obrázek by měl být použit jako ikona identifikující tuto sadu MIDletů
MIDlet-Description	Ne	Popis sady MIDletů
MIDlet-Info-URL	Ne	URL, kde je možné najít další informace o této sadě MIDletů
MIDlet-<n>	Ano	Jméno, ikona a název hlavní třídy n-tého MIDletu oddělené čárkami
MIDlet-Jar-URL	Ne	URL, odkud je možné stáhnout jar soubor této sady MIDletů
MIDlet-Jar-Size	Ne	Velikost JAR souboru
MIDlet-Data-Size	Ne	Minimální velikost paměti, kterou tato sada MIDletů potřebuje pro uložení trvalých dat. Výchozí hodnota je 0
MicroEdition-Profile	Ano	Požadovaný J2ME profil, např. MIDP-1.0
MicroEdition-Configuration	Ano	Požadovaná J2ME konfigurace, např. CLDC-1.0

Tabulka 2: Přehled všech předdefinovaných atributů sady MIDletů a jejich povinnost

Textový soubor JAD je deskriptor aplikace, který má stejný formát jako manifest soubor. Při instalaci sady MIDletů se nejprve přenese JAD soubor a pak podle informací v něm obsažených se nahrává soubor JAR. Následující atributy jsou povinné:

- MIDlet-Name
- MIDlet-Version
- MIDlet-Vendor
- MIDlet-Jar-URL
- MIDlet-Jar-Size

Kromě těchto povinných atributů lze do deskriptoru dopsat vlastní libovolné atributy, které začínají řetězcem MIDlet. K těmto atributům má programátor přístup pomocí metody getAppProperty (String key) třídy MIDlet [5].

2.4. Java Mobile Media API

Java Mobile Media API je volitelný balíček, který je určen pro práci s multimédii na platformě JavaME. Jeho obsahem jsou třídy a metody pro práci s videem, zvukem i obrázky a rozšiřuje základní možnosti médií, které jsou definovány v MIDP.

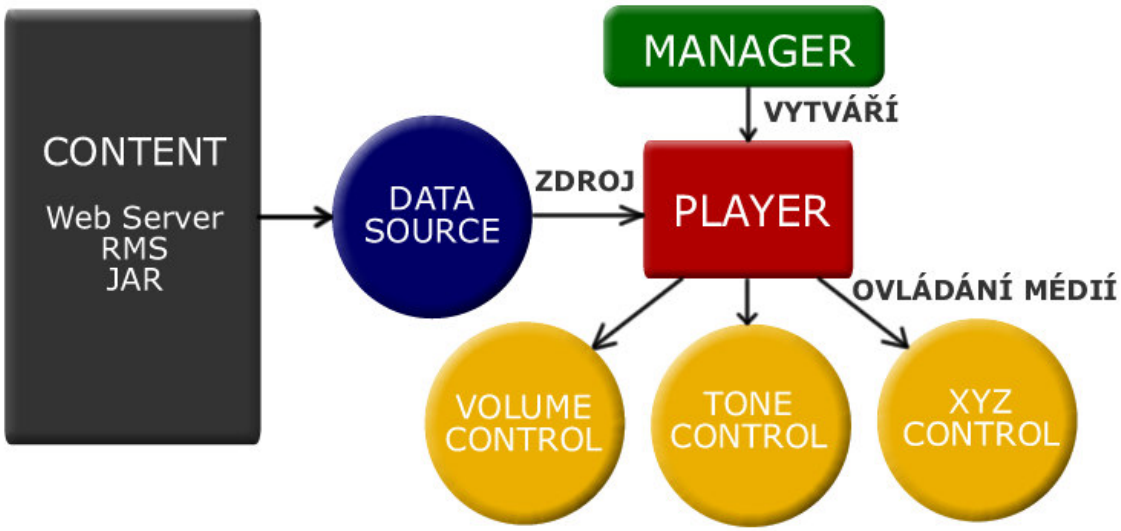
MMAPI vzniklo v roce 2001 jako specifikace JSR-135. Hlavním iniciátorem a podílníkem na vývoji byla firma Nokia. Na vývoji se podíleli odborníci z několika nejvýznamnějších společností na poli s mobilními telefony (Nokia, Siemens, Philips, Vodafone a další). První vydání MMAPI bylo v roce 2002. Aktuální verze MMAPI 1.2 je z května roku 2006.

Jako volitelný balíček lze MMAPI použít v kombinaci s libovolným profilem. Část MMAPI byla vložena do specifikace MIDP 2.0, kde tvoří základní práci se zvukem. MMAPI se skládá z několika balíčků obsažených v balíčku `javax.microedition`. Hlavní částí MMAPI je balíček `javax.microedition.media`, `media.Manager`, `media.Player`, `media.control`, `media.protocol`, `media.Palyerlistener`.

2.4.1. Architektura MMAPI

Třída Manager slouží jako vstup do celého API (Obr. 5). Manager je statická třída, která vytváří pouze statické metody. Aplikace tedy nemůže vytvořit instanci sama sebe. Manager poskytuje statické metody pro vytvoření instance Playeru a ke zjištění

podporovaných protokolů a typy medií, které je dané zařízení schopno přehrát. Poskytuje také metodu pro pohodlné vytváření jednoduchých tónů.



Obr. 5: Architektura MMAPI

2.4.2. Manager

K vytvoření instance Playeru slouží statická metoda třídy `Manager` `createPlayer`. Tato metoda může mít tři různé vstupní zdroje.

Metoda **`createPlayer(String locator)`** vytvoří Player pomocí locatoru ve tvaru URI, která popisuje daný zdroj. URI je definována tvarem:

`<protokol>:<část definovaná protokolem>`

Locator může být tvořen následovně:

- Locator pro přístup k souboru:
  `"file://"cesta k souboru`
- Locator pro přístup k RTP streamu
  `"rtp://"adresa[":" port] ["/" typ]`
  Adresa a port jsou dány RTP protokolem. Typ může být audio, video nebo text.
- Locator pro přístup k rádiovému vysílání
  `"capture://radio?" [parametry]`
  Parametry jsou nutné informace k příjmu rádia – tedy např. frekvence vysílání.

Metoda **`createPlayer(InputStream stream, String type)`** vytvoří Player pomocí řetězce InputStream popisujícího typ obsahu. Typ je definován jako registrovaný typ MIME.

Známé MIME typy:

- `audio/midi` – MIDI soubory
- `audio/mpeg` – Mp3 soubory
- `audio/x-wav` – WAV soubory
- `audio/aac` – AAC soubory
- `application/ogg` – OGG soubory
- `video/x-msvideo` – AVI soubory
- `video/mp4` – MP4 soubory
- `video/3gpp` – 3GP soubory

Pokud nebude parametr `typ` nastaven, tedy `null`, Manager se pokusí zjistit na základě dat MIME typ sám. Tato operace je netriviální a proto manager nemusí být schopen určit tento typ. V takovém případě skončí volání výjimkou `MediaException`. Také Player vytvořený pomocí `InputStreamu` může mít problémy s nastavením pozice při přehrávání, např. při volání metody `setMediaTime()`.

Metoda **`createPlayer(DataSource source)`** využívá abstraktní třídy DataSource. Pomocí její implementace lze před Playerem skrýt přístup k danému zdroji dat za jednotné rozhraní. Nedochází tak k potížím, jako při vytváření pomocí InputStreamu, při nastavování pozice při přehrávání média.

Třída Manager také nabízí možnost získání informací o podporovaných formátech a protokolech v daném zařízení. Tuto možnost poskytuje pomocí dvou statických metod a to `getSupportedContentTypes (String protocol)` a `getSupportedProtocols (String kontent_type)`. První vrací pole všech podporovaných formátů daného zařízení pro daný protokol uvedený v parametru. Pokud je parametr null vrátí pole všech podporovaných formátů pro všechny protokoly. Druhá metoda pak vrátí k danému formátu podporovaný protokol. Stejně jako u první metody i tady, pokud je parametr roven hodnotě null vrátí všechny podporované

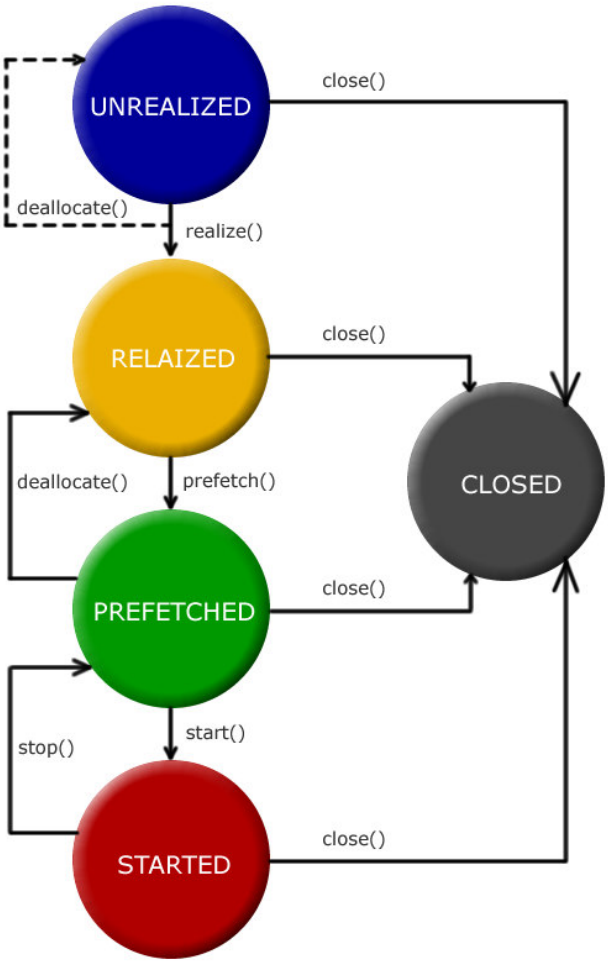
protokoly pro všechny formáty. Pokud ani jedna metoda nevrátí žádný výsledek (prázdné pole) znamená to, že je nepodporovaný nebo neznámý typ nebo protokol.

2.4.3. Player

Rozhraní Player se nachází v balíčku `javax.microedition.media`, reprezentuje třídy, které řídí vlastní přehrávání média. Pro každý podporovaný typ existuje specifický druh Playeru, který umí tento typ přehrát. Objekt typu Player se řídí daným životním cyklem. Životní cyklus Playeru má pět stavů (Obr. 6). Jsou to stavy *nerealizovaný (unrealized), realizovaný (realized), připravený (prefetched), běžící (started),* a *ukončený (closed).* Mezi těmito stavy se dá přecházet pomocí metod `realize()`, `prefetch()`, `start()`, `stop()`, `close()` a `deallocate()`. Aktuální stav Playeru se dá zjistit za pomocí metody `getState()`, která vrací hodnotu `UNREALIZED`, `REALIZED`, `PREFETCHED`, `STARTED` nebo `CLOSED`. Player je po vytvoření ve stavu `UNREALIZED`. Nejjednodušší spuštění Playeru je zavoláním metody `start()`, která spustí samotné přehrávání. Player prochází pak stavy *realizovaný* a *připravený* do stavu *běžící*. Z tohoto stavu po ukončení přehrávání přechází zpět do stavu *připravený*. Z každého stavu pak může přejít voláním metody `close()` do stavu *ukončený*.

Stav nerealizovaný

V tomto stavu se Player nachází po svém vytvoření třídou Manager a nachází se v něm do doby dokud není volána metoda `realize()`, která vykoná přechod do stavu *realizovaný*. Jakmile Player opustí stav *nerealizovaný* může být tento přechod ještě přerušen zavoláním metody `deallocate()`. Pokud ale dosáhne stavu *realizovaný* může se pohybovat jen mezi zbývajícími čtyřmi stavy. Ve stavu *nerealizovaný* nemá Player ještě načtené žádné informace o médiu a tudíž nemůže provádět metody, které jsou na těchto informacích závislé.



Obr. 6: Životní cyklus Playeru

Stav realizovaný

Do tohoto stavu přechází Player voláním metody `realize()`. Při volání této metody dochází k získávání informací o datech určených k přehrávání. Tato operace může být časově a paměťově náročná neboť Player přistupuje k samotným datům. V této fázi může také probíhat kontrola oprávnění k přehrávání těchto dat. Po ukončení této metody může Player využívat všechny metody rozhraní. Z toho stavu může Player přecházet do stavů *připravený* a *ukončený* pomocí metod `prefetch()` a `close()`.

Stav připravený

Do tohoto stavu se dostane Player zavoláním metody `prefetch()`. Toto volání může být podobně náročné jako volaní metody `realize()`, neboť Player přistupuje k exkluzivním zdrojům, které bude potřebovat pro spuštění. Také v této fázi probíhá kontrola oprávnění k přístupu k těmto datům. Player v tomto stavu však ještě nemusí vlastnit všechny prostředky. Při následném volání metody `start()` spustí

přehrávání v nejkratší možné době. Z toho stavu může Player přejít do stavů *realizovaný* a *ukončený*. Metoda `deallocate()` uvolní veškeré exkluzivní zdroje, které byly získány metodou `prefetch()` a Player se dostane do stavu *realizovaný*.

Stav běžící

Tento stav je klíčovým stavem životního cyklu. Probíhá v něm přehrávání média. Do toho stavu se dostane Player voláním metody `start()`, která opět kontroluje oprávnění k přístupu k datům a spustí přehrávání. Přehrávání média začíná implicitně na začátku média. Lze to však změnit nastavením času metodou `setMediaTime()`. Přehrávání končí buď koncem média nebo pomocí ovladače `stopTimeControl()`. Pak přechází do stavu *připravený*, do kterého může přejít i voláním metody `stop()`. Metoda `stop()` nepřesouvá čas přehrávání na začátek média, tudíž po opětovném volání metody `start()` pokračuje v přehrávání od místa, kde bylo zastaveno.

Stav ukončený

Do tohoto stavu může Player přejít z každého stavu voláním metody `close()`. Při tomto přechodu Player uvolní všechny zdroje, které dosud používal. Tento stav je konečný. Player může volat pouze metody `getState()` a `close()`. Volání jiných metod končí výjimkou `IllegalStateException`.

Kromě zmíněných metod pro přechod mezi stavy Playeru lze použít ještě několik dalších, které slouží pro práci s Playerem. Jsou to metody: `getDuration()`, `getTimeBase()`, `setTimeBase()`, `getMediaTime()`, `setMediaTime()` a `setLoopCount()`. Player dědí z třídy `Controllable` a nabízí tedy i její metody `getControl()` a `getControls()`.

Metoda `getDuration`() vrací hodnotu, která je rovna délce trvání média. Pokud není tato doba známa, třeba při volání ve stavu *nerealizovaný* nebo při přehrávání média, které nemá dobu učenu (přehrávání streamu nebo radia), vrátí metoda konstantu TIME_UNKNOWN. Délka je uváděna v mikrosekundách.

Metoda `getMediaTime`() a `setMediaTime`() pracují s aktuální pozicí přehrávaného média. Metoda `getMediaTime`() vrací hodnotu aktuální pozice přehrávání v mikrosekundách. Pomocí metody `setMediaTime`() je možné tento čas

změnit. Pokud tuto metodu zavoláme ve stavu *běžící*, provede skok v přehrávání média na určenou pozici. Pokud ji zavoláme v některém jiném stavu, nastaví pozici počátku přehrávání. Tato operace časových skoků může být časově náročná.

Metody `setTimeBase()` a `getTimeBase()` pracují s objekty rozhraní `java.microedition.media.TimeBase`. Objekt TimeBase představuje měřič času, který je součástí každé instance Playeru. Tyto metody jsou určeny hlavně pro synchronizace několika instancí Playerů. Může to vypadat následovně `player_video.setTimeBase(player_audio.getTimeBase)`.

Metoda `setLoopCount(int count)` nastaví počet opakování pro přehrávání média. Pokud je parametr nastaven na -1 je médium přehráváno donekonečna. Hodnota 0 není povolena a skončí výjimkou. Počet opakování musí být nastaven dřív než Player přejde do stavu *běžící*.

2.4.4. PlayerListener

Slouží pro zachycování událostí v MMAPI a nachází se v balíčku `javax.microedition.media`. Jednotlivým instancím Playeru může být přiřazen libovolný počet listenerů. Ty se pak starají o odposlouchávání všech událostí vzniklých při běhu Playeru. Jeden listener může být také přiřazen několika instancím Playeru. Nastavuje se voláním metody `addPlayerListener()` a odebírá se metodou `removePlayerListener()`. Umí spravovat jak standardní události Playeru tak uživatelsky definované.

Rozhraní PlayerListener obsahuje pouze jednu povinnou metodu, kterou musí implementující třída obsahovat a to `playerUpdate(Player player, String event, Object eventData)`. Tato metoda potom spravuje jednotlivé události. Jednotlivé parametry pak určují instanci Playeru, od které událost přišla, konkrétní událost a informaci, kterou s sebou můžou jednotlivé události nést. Kompletní seznam událostí je uveden v Tabulka 3.

Podle specifikace MMAPI je možné vytvořit si vlastní události. Pak je třeba odchytávání takové události listenerem v metodě `playerUpdate()` kontrolovat parametrem event. Definování takové události je však netriviální operace, která vyžaduje vlastní implementaci rozhraní Playeru.

2.4.5. Control

Rozhraní `javax.microedition.media.Control` je určené pro práci s ovladači média. Slouží k logickému sdružování konkrétních ovladačů. Jednotlivé ovladače jsou od sebe velmi odlišné, neobsahuje toto rozhraní žádné atributy ani metody. Ovladače jsou volitelnou součástí MMAPI. Proto nemusí být implementované pro každé zařízení. Avšak pro některé typy Playerů specifikace MMAPI definuje povinné a doporučené ovladače. Rozhraní těchto ovladačů se nacházejí v balíčku `javax.microedition.media.control`.

Popis jednotlivých rozhraní typu Control:

FramePositioningControl - ovladač pro správu pozic video rámců – při přeskakování na danou pozici

GUIControl - ovladač pro práci s GUI komponentami

MetaDataControl - ovladač k získání meta informací o mediu – informace jsou ve formě klíč-hodnota.

MIDIControl - ovladač pro správu zařízení pracujících s MIDI

PitchControl - ovladač pro správu výšky tónů implementovatelný pro MIDI nebo vzorkované audio formáty – pro snižování či zvyšování výšky přehrávaných tónů aniž by byla ovlivněna rychlost a hlasitost přehrávání.

RateControl - nastavuje rychlosti přehrávání media – nastavována v promilích.

RecordControl - řídí nahrávání dat aktuálně hraných Playerem.

StopTimeControl - nastavuje čas, ve kterém player zastaví přehrávání.

TempoControl - ovladač pro správu tempa u MIDI souborů.

ToneControl - ovladač určený k přehrávání uživatelsky definovaných sekvencí tónů.

VideoControl - ovladač nezbytný k přehrávání videa.

VolumeControl - nastavuje hlasitost přehrávání – v rozsahu 0–100.

Pro získávání objektů typu Control slouží rozhraní Controllable, které obsahuje metody getControl(String controlType) a getControls(). Metoda getControl vrací konkrétní ovladač nebo null, pokud není daný ovladač podporován. Metoda getControls() vrací všechny dostupné ovladače ve formě pole. Každý Player může přistupovat jen k určité skupině ovladačů, která je závislá na typu média. Metoda getControls() tedy vrátí jen dostupné ovladače z této skupiny pro konkrétní médium [1], [2], [4], [6].

Název události	Popis	eventData
BUFFERING_STARTED	voláno při startu bufferování pro zpracování media	čas startu bufferování
BUFFERING_STOPPED	ukončení bufferování	čas ukončení bufferování
CLOSED	Player dosáhla stavu *ukončený*	nemá přiřazeno eventData
DEVICE_AVAILABLE	uvolnění zdroje žádaného playerem	řetězec obsahující jméno dostupného zdroje.
DEVICE_UNAVAILABLE	pokud se zdroj žádaný playerem, stane nedostupným	řetězec obsahující jméno nedostupného zařízení
DURATION_UPDATED	pokud se dříve neznámá doba trvání stane známou	nový čas media.
END_OF_MEDIA	při dosažení konce media v aktuální smyčce přehrávání	čas, ve kterém bylo dosaženo konce media.
ERROR	vznik chyby	řetězec s chybovou hláškou
RECORD_ERROR	vznik chyby během nahrávání	řetězec s chybovou hláškou
RECORD_STARTED	voláno při startu nahrávání	čas, ve kterém bylo zahájeno nahrávání
RECORD_STOPPED	ukončení nahrávání	čas, ve kterém bylo zastaveno nahrávání
SIZE_CHANGED	voláno při změně velikosti videodisplaye	objekt typu VideoControl obsahující novou velikost
STARTED	voláno při dosažení stavu *běžící*	čas, ve kterém bylo spuštěno přehrávání
STOPPED	voláno při zastavení přehrávání voláním metody stop()	čas, ve kterém bylo zastaveno přehrávání
STOPPED_AT_TIME	zastavení přehrávání v čase dříve definovaném metodou setStopTime() ovladače StopTimeControl	čas, ve kterém bylo zastaveno přehrávání.
VOLUME_CHANGED	voláno při změně hlasitosti	VolumeControl obsahující hlasitost

Tabulka 3: Standardní události Playeru

3. Kódování multimédií

3.1. Video kódování

Video sekvence se skládá z mnoha statických snímků jdoucích po sobě. Komprese takových snímků je založena na metodách redukujících nadbytečné a vjemově zbytečné části video sekvence. Tato nadbytečnost může být kategorizována z několika hledisek a to prostorového, časového a spektrálního. Prostorová nadbytečnost odkazuje na korelaci mezi sousedními pixely. Časová nadbytečnost zase znamená, že stejný objekt nacházející se v předchozím obrazu je i v obrazu aktuálním. Spektrální nadbytečnost je korelace mezi různými barvami objektů v obrazu. Tato nadbytečnost může být redukována zavedením kompenzace pohybu, která popisuje pohyb mezi aktuálním a předchozím snímkem. Můžeme tedy říct, že aktuální snímek je predikován z předchozího.

Často nám nestačí potlačit pouze nadbytečnost v sekvenci, ale musíme zahodit také užitečné informace obrazu. Kodéry se tedy snaží využít nedokonalosti lidského vnímání. Snaží se zahodit informaci, která je nedůležitá pro subjektivní vnímání obrazu. Takto vytvořený bitový tok je kódován efektivním bezeztrátovým kódem. Pro toto kódování se používají hlavně kódy s proměnnou délkou.

3.1.1. H.263

H.263 je kodek používaný v různých multimediálních službách. Je určen pro video přenosy se stálou bitovou rychlostí. Původně byl vyvíjen hlavně pro přenos video konferenčních hovorů. Pro multimédia v mobilních telefonech byl definován jako závazný profil 0, úroveň 10 (Profile 0, Level 10). Toto doporučení je dodržováno většinou dnešních multimediálních terminálů. Kodek používá diskrétní kosinovou transformaci pro redukci prostorové nadbytečnosti. Tato transformace převádí bloky pixelů na koeficienty, které reprezentují prostorovou frekvenci jednotlivých částí bloku. Pouze frekvence nacházející se v bloku mají nejvyšší hodnotu koeficientu, ostatní se blíží k nule. Například pokud máme jednobarevný blok, tak po transformaci bude pouze jeden DCT koeficient různý od nuly, zbytek bude roven nule. Nejvyšší koeficienty jsou poté ještě kvantovány a kódovány pomocí run-lenght kódů. H.263 poskytuje 31 kvantizačních kroků. Čím méně je použito kvantizačních hladin, tím více dochází k degradaci obrazu.

3.1.2. MPEG-4 Visual

Je to kodek, který je velmi rozšířen v dnešních telefonech Nokia a dalších mobilních telefonech. Stejně jako H.263 i MPEG-4 Visual obsahuje několik profilů pro různé účely. Platforma firmy Nokia podporuje Simple Profile 0. MPEG-4 Visual je založený také na diskrétní kosinové transformaci. Nepohyblivé části textury jsou kódovány waveletovou transformací.

3.2. Audio kódování

Všechny zvuky mohou být reprezentovány sumou vlnění o různých frekvencích a amplitudách. Zvuky jsou digitalizovány pomocí vzorkování analogového zvuku. Pro normální zvuky je vhodné použít vzorkovací frekvenci 44,1 kHz, aby byla zajištěna vysoká kvalita. Pro řeč dostačuje vzorkovací frekvence 8 kHz. Zvuk může být komprimován různými cestami. Nejjednodušší metoda je požití adaptivního kvantizačního kroku na zvukové vzorky (např. ADPCM). Další může být třeba použití logaritmického kvantizačního kroku (např. A-law PCM). Složitější metody využívají nedokonalosti lidského vnímání. Část audio signálu může být zahozeno nebo komprimováno. Pokročilé metody kódování jsou rozděleny na kódování obecných zvuků a kódování řeči. Obecné kódování je zaměřeno hlavně na hudbu a stejně tak jako na řeč. Kódování řečové je zaměřeno pouze na zakódování mluveného slova.

3.2.1. AMR-NB

AMR (Adaptive Multi-Rate) je jeden z dnes nejpokročilejších kódovacích standardů řeči. Byl vyvinut organizací ETSI (European Telecommunications Standards Institute). Obsahuje osm kódovacích módů od bitového toku 4,75 do 12,2 kb/s. AMR s bitovým tokem 12.2 kb/s je stejný jako řečový kodek EFR (Enchanced Full-Rate). AMR je kodek typu mono.

3.2.2. MP3

MPEG-1 Audio Layer, známý jako MP3, je ztrátový formát, který používá mnoho technik k redukci velikosti audio dat. Zahazování audio informací je založeno na lidském psychoakustickém modelu. Tento formát je poskytován mnoha dnešními zařízeními, jako jsou telefony, audio přehrávače a počítače.

3.2.3. AAC

AAC (MPEG-4 Advanced Audio Cosiny) je nástupce MP3. Má mnoho výhod oproti formátu MP3 v efektivnosti kódování a práci s frekvencemi. Můžeme říct, že 96 kb/s u AAC je lepší nebo minimálně srovnatelné se 128 kb/s u MP3. I když má tento formát lepší parametry, není tak oblíbený jako již zmíněný formát Mp3.

3.3. Dnes nejvíce používané video formáty

- AVI – Microsoft Audio-Video Interleaved
- MOV – Apple QuickTime formát
- MPG – MPEG-1 formát
- 3GP – 3GPP formát
- MP4 – MP4 formát

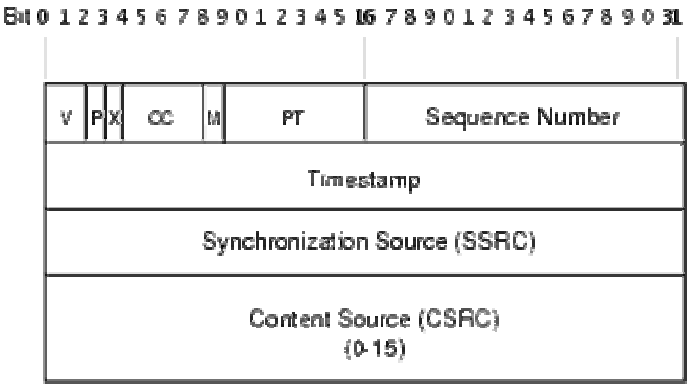
Většina dnešních mobilních telefonů, které jsou schopny přehrávat video soubory, podporuje formát 3GPP (přípona .3gp). V jednotlivých videoformátech jsou rozdíly v kódování videa. [3]

4. Přenos multimédií přes síť

Multimédia můžeme přes síť přenášet dvojím způsobem. Prvním je přenos offline. To znamená, že si multimediální soubor stáhneme do počítače, nebo jiného zařízení, a po té si ho přehrajeme z uloženého umístění. Druhým způsobem nemusíme multimédia ukládat na datové médium zařízení. Přehráváme ho takzvaně online. Multimediální soubor tak není uložen na lokálním úložišti zařízení. K tomuto účelu bylo vyvinuto několik aplikačních protokolů: RTP (Real-Time Transport Protocol), RTCP (RTP kontrol Protocol) a RTSP (Real-Time Streaming Protocol).

4.1. RTP (Real-Time Transport Protocol)

RTP je protokol aplikační vrstvy. Definuje standardní paketový formát pro doručování zvukových a video dat přes síť (internet). Byl vyvinut skupinou Audio-Video Transport Working Group pod IETF (Internet Engineering Task Force) jako standard RFC 1889 a poprvé publikován v roce 1996. Tento standard byl rozšířen v roce 2003 jako specifikace RFC 3550. Protokol je hojně využíván v systémech pro streamování médií jako je telefonování, videokonference a v push-to-talk aplikacích. Často se RTP protokol využívá ve spojení s dalšími protokoly jako jsou H.323, SIP (Session Initiation Protocol), MGCP (Media Gateway Control Protocol) nebo SCCP (Skinny Call Control Protocol). RTP je také základem pro VoIP (Voice over Internet Protocol) technologie. RTP protokol je nejčastěji využíván ve spojení s protokolem RTCP, kdy RTP protokol přenáší multimediální data a protokol RTCP je používán pro monitorování přenosových statistik a zajištění QoS (Quality of Service). Pokud jsou tyto protokoly používány současně vysílá a přijímá protokol RTCP na portu o jeden vyšší než aktuálně využívá RTP protokol. RTP protokol podporuje jak unicast (vysílání k jednomu příjemci) tak multicast (skupinové vysílání) přenosy. RTP negarantuje doručitelnost paketu, avšak segmentace dat umožňuje detekci chybějících paketů. RTP je doporučován jako primární protokol na doručování audio/video obsahu po IP sítích. Multimediální aplikace nejsou náchylné na ztrátu paketů, ale musí být doručovány ve správném pořadí. Přenos není náchylný na zpoždění, ale na kolísání zpoždění ano. RTP protokol je definován jak pro přenos přes TCP tak i UDP protokoly. V praxi se nejvíce používá pro doručování UDP protokol.



Obr. 7: Hlavička RTP paketu

Informace v hlavičce říkají příjemci jak má rekonstruovat přijatá data a jak jsou obsažená data skládána (paketizována). Minimální velikost hlavičky paketu je 12 bajtů. Po hlavičce může následovat volitelné rozšíření hlavičky.

Význam jednotlivých polí v hlavičce RTP protokolu je následující:

- V 2 bity – verze protokolu (aktuální verze je 2)
- P 1 bit – Padding – specifikuje jestli je na konci paketu použito nějakých extra bitů (např. pro šifrování)
- X 1 bit – eXtension – indikace volitelné hlavičky
- CC 4 bity – počet identifikátorů CSRC, které následují po hlavičce
- M 1 bit – značí jestli je použit Marker v rozšířeném záhlaví
- PT 7 bitů – Payload Type – identifikuje formát RTP jako je kompresní schéma nebo kódování
- SN 16 bitů – číslování paketů – inkrementováno při odesílání paketů, užíváno příjemcem k detekci ztracených paketů
- TS 32 bitů – používáno k přehrávání přijatých vzorků ve správných intervalech, při příjmu více stop použito k synchronizaci
- SSRC 32 bitů – jednoznačný identifikátor zdroje
- CSRC 0 až 15 položek, každá 32 bitů – identifikátor pokud je použito několik různých zdrojů současně [7], [8], [9]

4.2. RTCP (RTP Control Protocol)

RTCP je řídící protokol pro přenos zvuku a videa v reálném čase sítí. Doplňuje protokol RTP. Je definován ve specifikaci RFC 3550, která nahrazuje RFC 1889. Poskytuje řídící informace pro protokol RTP. Sám však žádné data nenese. Používá se k pravidelnému přenosu kontrolních paketů účastníků relace. Hlavní funkcí je zajištění

kvality služeb (QoS), monitorování a řízení toku dat. RTCP shromažďuje data o spojení jako například počet ztracených paketů, kolísání zpoždění, počet odeslaných paketů a bajtů. Aplikace pak mohou tyto informace využít ke zkvalitnění služeb například omezení datového toku nebo použitím jiného kodeku. RTCP používá stejně jako RTP pro přenos dat UDP protokol s číslem portu o jedno vyšší než RTP.

RTCP definuje pět typů zpráv:

Sender Report (SR)

Účastníci spojení pravidelně posílají zprávy typu Sender report. Sender Report zprávy obsahují informace o probíhající komunikaci a přenášejí také přijímací statistiky pro všechny posílané pakety RTP. Jinými slovy umožňují přijímači odhadovat komunikační rychlost a kvalitu přenosu. Sender report obsahuje časová razítka (timestamps), ve kterých je uloženo číslo v sekundách od půlnoci 1. ledna 1970. Úplné časové razítko umožňuje přijímači synchronizovat různé typy RTP zpráv. Tato skutečnost je obzvláště důležitá, když jsou přenášena audio a video data (jednotlivé streamy používají vzájemně oddělená časová razítka).

Receiver Report (RR)

Receiver Report slouží pro pasivní účastníky spojení. Tedy pro stanice, které neodesílají žádné RTP pakety. Zprávy informují odesílatele a ostatní příjemce o kvalitě služeb, problémech přijímačů a dále obsahují čísla ztracených paketů a informaci o kolísání zpoždění (jitter) u přijímače. Výsledkem může být například to, že vysílající aplikace sníží nebo naopak zvýší kvalitu poskytovaného obsahu.

Source Description Message (SDES)

Vysílač dat pravidelně posílá zprávy nesoucí informace o sobě, které jsou k dispozici ostatním uživatelům. Podle RFC 1889 jsou typy SDES zpráv: END (konec SDES seznamu), CNAME (kanonické jméno), NAME (jméno uživatele), EMAIL (emailová adresa), PHONE (telefonní číslo), LOC (geografická poloha), TOOL (název aplikace, která generuje RTP provoz), NOTE (zpráva, která popisuje současný stav vysílače dat) a PRIV (aplikační a experimentální rozšíření).

Goodbye Message (BYE)

Vysílač dat posílá BYE zprávu k ukončení spojení. Ačkoli ostatní zdroje mohou sami zjistit nepřítomnost jiného zdroje, tato zpráva tuto událost

přímo oznamuje. Znalost této události je velice užitečná pro zařízení zvané mixer. Pokud mixer přijme BYE paket, tak ho v nezměněné podobě posílá ostatním uživatelům.

Application-Specific Message (APP)

Application-Specific slouží pro zasílání zpráv, které nejsou definované ve standardu a umožňuje tak definici nových typů zpráv. APP pakety se používají také pro experimentální účely. [9]

4.3. RTSP (RTP Streaming Protocol)

RTSP je síťový řídící protokol, který umožňuje vzdálené řízení streamovacího serveru. Je používán pro sestavení spojení a řízení médií mezi koncovými účastníky. Poskytuje příkazy pro řízení přehrávání médií ze serveru jako je *play*, *pause*, *revind*. Spolupracuje s protokoly pro přenos médií RTP a RSVP (Resource Reservation Protocol) k zajištění kompletního servisu pro uživatele. RTSP protokol je velmi podobný protokolu HTTP (HyperText Transfer Protocol). Standardní port na kterém RTSP pracuje je 554. RTSP byl vyvinut skupinou Multiparty Multimedia Session Control Working Group v roce 1998 a publikován jak RFC 2326. Průběh komunikace je vidět na Obr. 8.

RTSP definuje několik typů zpráv:

DESCRIBE

Součástí je RTSP adresa (rtsp://…) a typ dat, které mohou být ovládány

SETUP

Specifikuje jak mají být jednotlivé proudy přenášeny. Musí být poslána dříve než zpráva PLAY

PLAY

Začíná přehrávat jeden nebo více proudů.

PAUSE

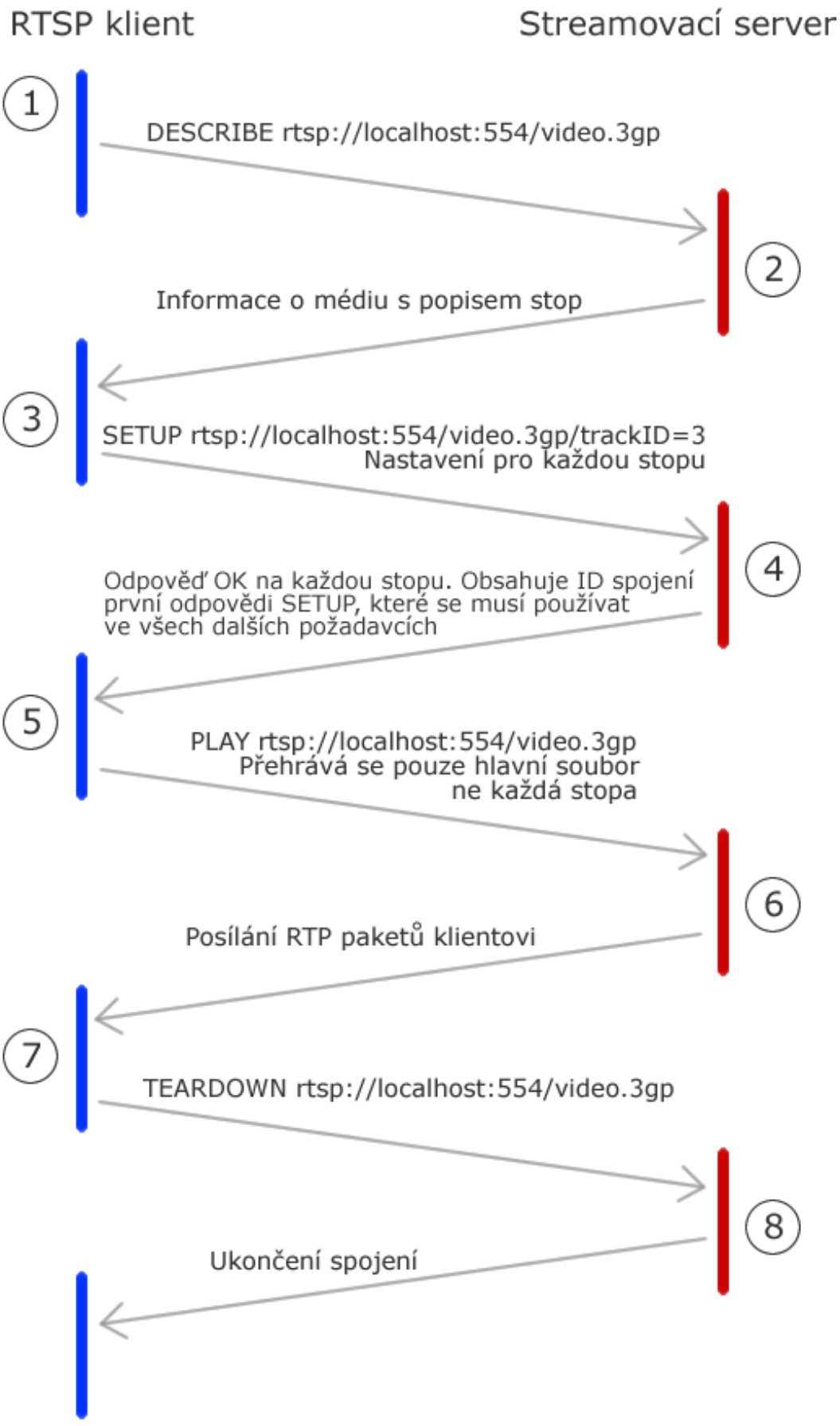
Dočasně pozastavuje přehrávání. Pomocí PLAY přehrávání opět pokračuje od místa pozastavení

RECORD

Může být použit pro ukládání proudu na server.

TEARDOWN

Používá se k přerušení spojení. Zastaví všechna přehrávaná média a uvolní prostředky na serveru. [9], [10]



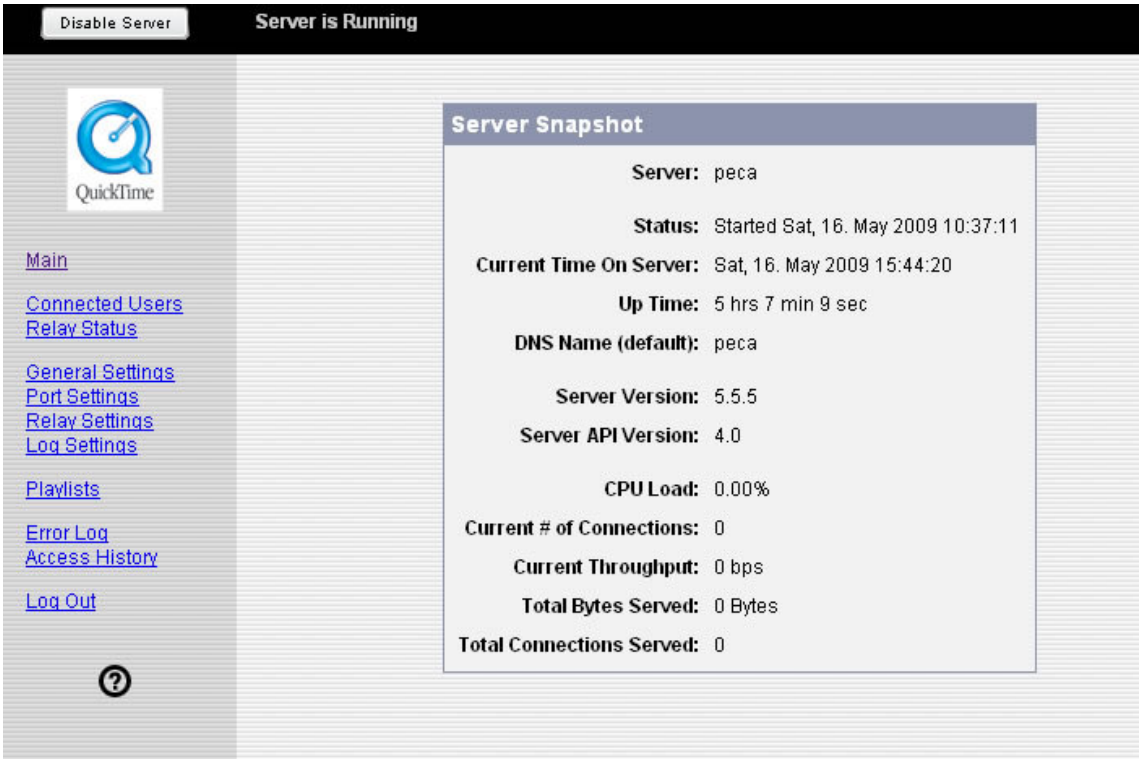
Obr. 8: Průběh RTSP komunikace

5. Technologie použité na serverové části

Součástí této práce je jak klientská část tak serverová část. Serverová část řešení zahrnuje několik technologií, které jsou propojeny, aby celé řešení správně pracovalo. Na serverové části je zprovozněn streamovací server Darwin Streaming Server od firmy Apple, dále webový server Apache s podporou PHP a databázový systém MySQL.

5.1. DSS (Darwin Streaming Server)

Vyvinut firmou Apple. Je volně šiřitelným ekvivalentem QuickTime Streaming Server a je založen na jeho kódu. První verze byla vydána v roce 1999. Je to první server s otevřeným kódem pro streamování přes RTP a RTSP. Dokáže nativně streamovat velké množství mediálních typů včetně H.264, MPEG-4 Part 2 a 3GP. Administrace serveru probíhá pomocí webového prohlížeče na portu 1220 (viz.Obr. 9). Tento server používá i takový gigant jako YouTube pro stremování videa do mobilních zařízení. [11]



Obr. 9: Administrační rozhraní DSS

5.2. MySQL

MySQL je databázový systém vytvořený švédskou firmou MySQL AB. Je k dispozici pod bezplatnou licencí GPL i pod komerční licencí. Je multiplatformní

databáze. Komunikace s databází probíhá pomocí jazyka SQL. V PHP jsou zakomponovány funkce, které si velmi dobře rozumí s databází SQL. Hlavními soupeři MySQL jsou PostgreSQL, Microsoft SQL Server a Oracle. MySQL má mnoho výhod, včetně velkého výkonu, nízké ceny, snadného nastavení a naučení se jí. Je přenositelná a jsou k dispozici její zdrojové kódy. Je však vhodná pro „jednodušší“ projekty, pro složité informační systémy je výhodnější použití některého z konkurentů jako je PostgreSQL nebo Oracle.

5.3. Apache HTTP Server

Apache je softwarový webový server s otevřeným kódem. Lze ho používat na mnoha platformách (Linux, BSD, Microsoft Windows a další). Apache podporuje velké množství technologií pomocí tzv. modulů.. Některé jsou implementovány přímo jako zkompilované moduly, které rozšiřují funkčnost jádra systému. Apache podporuje mnoho skriptovacích jazyků jako je Perl, Python, TCL a PHP. Podporuje autentizační metody SSL a TLS. Umožňuje použití přepis URL adresy nebo také filtrování a mnoho dalších funkcí. Apache také podporuje virtuální hosting. Je primárně určen pro ovládání jak statických stránek, tak dynamického obsahu. Apache je nejpopulárnějším nasazovaným serverem na internetu. [12]

5.4. PHP (PHP: Hypertext Preprocessor)

Původně Personal Home Page je skriptovací programovací jazyk určený pro programování dynamických webových stránek. Začleňuje se většinou přímo do struktury HTML dokumentu. PHP je v současnosti velmi rozšířená technologie, která umožňuje snadné programování na straně serveru (tzv. server-side programming). Tedy skript napsaný v PHP je proveden na straně serveru podle zadaných kritérií a výsledek je odeslán klientovi stejným způsobem jako když se odesílá statická (X)HTML stránka. Po načtení klientem není možné stránku dále měnit. Stránku, kterou chceme měnit bez opětovného načtení ze serveru se provádí skripty na straně klienta – například pomocí JavaSkriptu. PHP umožňuje procedurální i objektově orientované programování. Typický PHP skript tedy obsahuje jednak části normálního (X)HTML kódu, a jednak části programového kódu. Když potom server obdrží požadavek od klienta na zpracování takového skriptu, vezme:

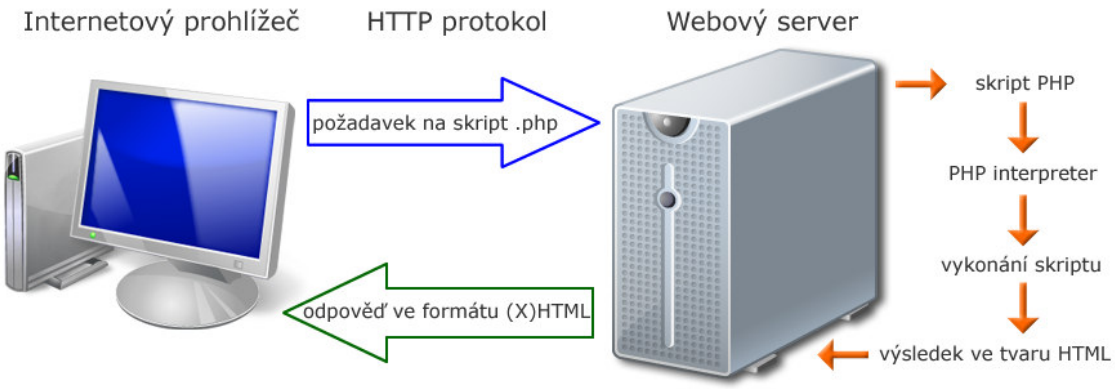
- části HTML kódu tak, jak jsou
- části PHP programového kódu provede

- výsledek pak zkombinuje a odešle prohlížeči

Tato filozofie je velmi mocná. Server totiž může provést jednu nebo dokonce několik operací a výsledek poslat do prohlížeče jako normální HTML stránku. Hlavní výhoda spočívá v tom, že díky tomuto přístupu je takový skript zcela nezávislý na použitém prohlížeči. Odpověď je odeslána již ve formě HTML výstupu, která je srozumitelná všem internetovým prohlížečům. Veškerá práce spojená s vykonáním skriptu je tedy závislá pouze na webovém serveru. PHP je jazyk nezávislý na platformě a skripty fungují bez úprav na mnoha různých operačních systémech. Dnes máme k dispozici verze pro operační systémy Windows, Unix i Linux. Obsahuje rozsáhlé knihovny funkcí pro zpracování textu, grafiky, práci se soubory, přístup k většině databázových serverů (mj. MySQL, ODBC, Oracle, PostgreSQL) a podporu celé řady internetových protokolů (HTTP, SMTP, SNMP, FTP, IMAP, POP3, LDAP a další).

5.4.1. Jak PHP funguje

Na Obr. 10 je vidět jak vypadá komunikace mezi uživatelem (prohlížečem webových stránek) a serverem poskytujícím překlad požadovaných skriptů.



Obr. 10: Komunikace internetového prohlížeče se serverem

Internetový prohlížeč (např. Internet Explorer, Firefox, Opera a další) vyšle požadavek na zobrazení dokumentu – skriptu PHP. Požadavek dorazí k http serveru, ten zjistí (podle toho jak je nakonfigurovaný), že dokument je skript a proto ho pošle PHP interpretovi. PHP interpret vykoná skript a výsledek běhu skriptu (většinou to bude v HTML formátu) vrátí HTTP serveru . Ten odešle výsledek zpět do internetového prohlížeče, který ho zobrazí uživateli, který sedí za počítačem. Ten ani netuší, že během zpracování vstupuje do procesu nějaký PHP interpret a stránka se mu jeví jako běžná stránka na internetu.

5.4.2. Výhody a nevýhody jazyka PHP

Výhody PHP:

- je relativně jednoduché na pochopení (podobné jiným jazykům)
- má syntaxi velmi podobnou jazyku C a je tedy většině vývojářů dost blízký
- podporuje širokou řadu souvisejících technologií, formátů a standardů
- je to otevřený projekt (open-source) s rozsáhlou podporou komunity
- existuje mnoho hotových kódů k okamžitému použití nebo funkční PHP aplikace. Velká část hotových kódů je šířena pod nějakou svobodnou licencí proto se dá bez problémů použít ve vlastních projektech
- dobře si rozumí s webovým serverem Apache (je to totiž sesterský projekt spravovaný Apache software foundation)
- snadno komunikuje s databázemi, jako je MySQL, PostgreSQL a řadou dalších
- je multiplatformní a lze jej provozovat s většinou webových serverů a na většině dnes existujících operačních systémech
- PHP podporuje mnoho existujících poskytovatelů webhostingových služeb

Nevýhody PHP, nebo lépe věci s kterými je nutno počítat při používání PHP, protože nevýhody v podstatě nejsou.

- PHP je interpretovaný, ne kompilovaný jazyk
- Kdokoli, kdo má přístup k serveru, může nahlédnout do skriptů uložených na serveru
- PHP je stále aktivně rozvíjeno proto se mohou některé funkce v budoucnu změnit nebo se můžou chovat jinak než dosud [13]

5.5. XHTML (Extended HyperText Markup Language)

Je značkovací jazyk pro tvorbu hypertextových dokumentů v prostředí WWW vyvinutý konsorciem W3C. Je následníkem jazyka HTML, jehož vývoj byl ukončen, a na rozdíl od svého předchůdce se jedná o aplikaci XML.

5.5.1. Verze

XHTML 1.0

První specifikace, jejíž cílem bylo převedení staršího jazyka HTML tak, aby vyhovoval podmínkám tvorby XML dokumentů a přitom byla zachována zpětná kompatibilita. Existuje ve třech druzích: *Strict*, *Transitional* a *Frameset*. Jediným podstatným rozdílem

proti HTML je, že veškeré prvky jsou párové a musí být uzavřeny. Hodnoty všech atributů musí být zapsány v uvozovkách.

5.5.2. Modularizace XHTML

Dalším krokem ve vývoji XHTML byla modularizace s cílem dosáhnutí vyšší flexibility napříč uživatelskými aplikacemi (WWW prohlížeče, mobilní zařízení, tiskárny, čtečky apod.).

XHTML Basic

Příklad minimální sady modulů potřebné k vytvoření XHTML dokumentu, která je cílená na mobilní aplikace.

XHTML Mobile Profile

Někdy také XHTML MP je postaveno na základě XHTML Basic a je určeno pro použití v mobilních telefonech. Někdy je také označováno jako WAP 2.0

XHTML 1.1 - modulově založené XHTML

Příklad rozsáhlé sady modulů pro komplexnější tvorbu XHTML dokumentů.Vynechává již prakticky všechny prezentační vlastnosti. Je velice podobné XHTML 1.0 Strict, ale narozdíl od něj může vzhledem ke své modularizaci sloužit jako základ budoucím rozšířeným dokumentům z rodiny XHTML.

XHTML-Print

Vývojové stádium konsorcia W3C (*Candidate Recommendation*). Zaměření na tiskový výstup.

XHTML 2.0

Vývojové stádium konsorcia W3C (*Working Draft*). Není zamýšleno tak, aby bylo zpětně kompatibilní se svými předchůdci.

5.6. CSS (Cascading Style Sheets)

CSS, neboli tabulky kaskádových stylů, slouží k popisu prezentace dokumentu napsaného v některém z jazyků HTML, XHTML nebo XML. Tento jazyk navrhla standardizační organizace W3C. Zatím byly uvolněny dvě verze specifikace CSS1 a CSS2 (CSS 2.1), usilovně se pracuje na verzi CSS3. Hlavním důvodem proč byly kaskádové styly vytvořeny je, aby se dal oddělit vzhled stránek od jejich struktury a obsahu. Původně to měl umožnit již jazyk HTML, ale ten byl značně ovlivněn konkurenčním bojem výrobců prohlížečů, kteří implementovali mnoho formátovacích

značek přímo do jazyka HTML. Byl to také důsledek nedostatečných standardů. Starší verze HTML obsahují celou řadu elementů, které nepopisují strukturu a obsah, ale i způsob zobrazení. Z hlediska vyhledávání a zpracování dokumentů je to nežádoucí. Proto nastupují tabulky kaskádových stylů, které tento problém řeší.

5.6.1. Výhody a nevýhody jazyka CSS

Hlavní výhodou kaskádových stylů je, že kód a obsah webu je uložen v souboru .htm a veškeré formátování a design je uložen většinou v externím souboru .css, který bývá společný pro celou web stránku. Takže pokud chceme změnit design celé stránky stačí změnit pouze soubor s kaskádovými styly. Mohou také existovat různé kaskádové styly pro různá výstupní zařízení. Například jiný styl pro obrazovku, jiný pro tiskárnu nebo pro PDA. Kaskádové styly myslí i na postižené a dovolují napsat styly pro hlasový syntezátor nebo hmatovou čtečku Braillova písma. V praxi přináší hlavně tyto možnosti:

- širší formátovací možnosti
- snadná tvorba a údržba stejného vzhledu celé stránky
- oddělení struktury a stylu
- dynamická práce se styly
- formátování XML dokumentů
- větší kompatibilita s alternativními prohlížeči
- kratší doba načítání stránky

Hlavní nevýhodou kaskádových stylů je stále nedostatečná podpora na straně majoritních prohlížečů. Různé prohlížeče interpretují styly různým způsobem nebo dokonce úplně špatně. Proto je obtížné napsat styly tak, aby se výsledek zobrazil na všech prohlížečích stejně. Situace se trochu zlepšila s příchodem prohlížeče Internet Explorer 7, jehož předchůdce je častým zdrojem problémů. Snad nový Internet Explorer 8 již bude bez výjimky tato pravidla dodržovat.

5.6.2. Koncepce jazyka

Stylový předpis se skládá z posloupnosti pravidel. Každé pravidlo určuje vzhled některého elementu (prvku) dokumentu, nebo skupiny elementů. Pravidlo začíná tzv. selektorem (ukazatelem), který specifikuje (adresuje) skupinu elementů. Selektor je následován seznamem deklarací, které určují vzhled vybrané skupiny elementů. Celý seznam je uzavřen ve složených závorkách. Jednotlivé deklarace jsou odděleny středníkem. [14]

6. Vypracování požadovaného systému

6.1. Zadání

Cílem této práce je vytvoření aplikace v JavaME, pro přehrávání multimediálních souborů, které jsou uloženy v mobilním telefonu nebo na kartě telefonu. Aplikace také bude umět přehrávat multimediální soubory uložené na vzdáleném serveru, takzvané video na vyžádání (VoD – Video on Demand). Pro přehrávání videa na vyžádání bude používat protokolů RTP, RTCP a RTSP.

Přehrávač bude umět přehrávat audio a video média jak lokálně tak vzdáleně. Bude mít základní ovládací prvky pro ovládání multimédií. Aplikace může využívat specifikace CLDC 1.1 a MIDP 2.0.

Na straně serveru bude databáze nahrávek, která bude ovládána pomocí PHP. Také musí být zprovozněn streamovací server, který bude poskytovat multimediální data pro klientskou aplikaci.

6.2. Vypracování klientské aplikace

Celý program je vyvíjen pomocí JavaME nad profilem MIDP 2.0 a konfigurací CLDC 1.1, podle zadání, ve vývojovém prostředí Netbeans 6.1 s Mobility Packem od firmy Sun Microsystems. Testování programu probíhalo pomocí vestavěného emulátoru Sun Java(TM) Wireless Toolkit 2.5.2 for CLDC. Později testován na emulátoru S60 3rd Edition SDK for Symbian OS, Feature Pack 2 v1.1, který je určen pro vývoj aplikací pod operačním systémem Symbian na mobilních terminálech značky Nokia, která tento emulátor i vyvíjí. Konečné testování a nasazení proběhlo na mobilním telefonu Nokia E51.

Přehrávač umí přehrávat video soubory, které jsou podporovány telefonem. To znamená, že jeho implementací nejsou žádné dodatečné kodeky, které by rozšiřovaly možnosti mobilního telefonu. Přehrávač disponuje základním grafickým uživatelským rozhraním. Přehrávač má základní ovládací prvky pro přehrávání multimédií jako jsou Pauza, Play, Stop, Mute. Jednotlivé soubory jsou otvírány pomocí volitelného balíčku `org.netbeans.microedition.lcdui.pda.FileBrowser`, který je součástí vývojového prostředí Netbeans 6.1, který však není plně podporován ve všech mobilních telefonech, ale většina novějších telefonů tento přístup k souborovému systému podporuje. Zvolil jsem tuhle možnost z důvodu uživatelské přívětivosti, neboť jinak by musel uživatel cestu k souboru zadávat ručně. Další možností by bylo využít

API jednotlivých výrobců, ale to by také nebylo příliš vhodné, protože aplikace by se tím stala méně přenosnou a pro každé zařízení by se musela upravovat.

Na rozdíl od Java aplikací pro běžné počítače, které pro tvorbu GUI mohou využívat bohaté nabídky komponent SWINGu či AWT, jsou aplikace vytvářené pro mobilní telefony v mnohém omezeny. Mohou zobrazovat v jednom okamžiku pouze jedno okno aplikace. Proto byla vytvořena speciální vysokoúrovňová API pro vytvoření uživatelského rozhraní, která vyhovují omezeným možnostem mobilních zařízení i požadavkům přenositelnosti. Jsou spíše zaměřena na funkcionalitu, než design aplikace. Pro každý přístroj je pak vzhled mírně odlišný. Kromě tohoto vysokoúrovňového API Java poskytuje i nízkoúrovňové API. V tomto případě je displej považován za plátno, na které se dá cokoliv nakreslit. Nabízí tak programátorovi široké možnosti, avšak práce s ním je mnohonásobně náročnější.

Přehrávač využívá vysokoúrovňové API, které pro tento účel aplikace dostačující. Aplikace se skládá ze základní obrazovky (Obr. 11). Po stisku tlačítka pod nápisem Volby se zobrazí základní menu aplikace (Obr. 12). Zde můžeme vidět možnosti Otevřít pro otevření souboru multimediálního typu nebo playlistu pro streamování videa z vlastního serveru, Otevřít stream pro zadání jakéhokoliv rtsp odkazu ve tvaru rtsp://server:port/video.3gp. Tento stream však musí být přizpůsoben pro přehrávání na mobilních zařízeních. Volba Media Info slouží pro informace o protokolech a typech multimédií, které je schopné přehrávač přehrát. Poslední položkou v menu je položka Help, ve které je krátká nápověda.

Obr. 11: Základní obrazovka

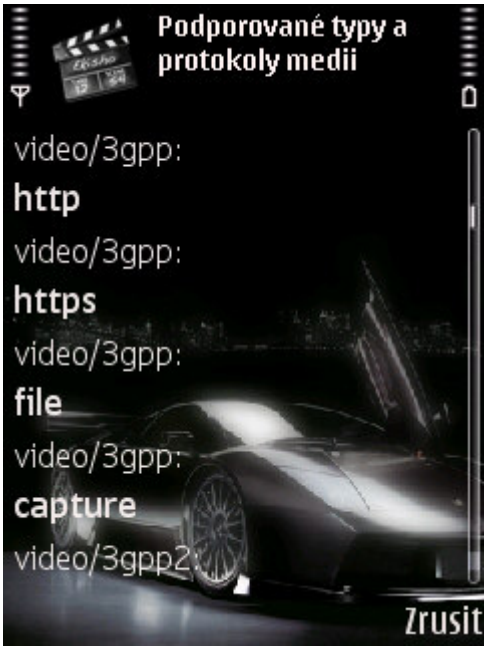
Obr. 12: Základní menu aplikace

Další obrazovkou, která se zobrazí po zvolení možnosti Otevřít je souborový manažer (viz. Obr. 13), kde procházíme jednotlivými úložišti a adresáři pro nalezení souboru, který chceme přehrát. Můžeme zvolit buď přímo multimediální soubor, který máme uložen na lokálním úložišti, nebo otevřít soubor s příponou .csv, kde máme vytvořen playlist, který si můžeme stáhnout ze serveru. Pro vzdálené přehrávání multimediálních souborů můžeme též zvolit položku v menu Otevřít stream (Obr. 14), kde můžeme přímo zadávat adresu serveru a videa, které chceme přehrát. Další možností menu je

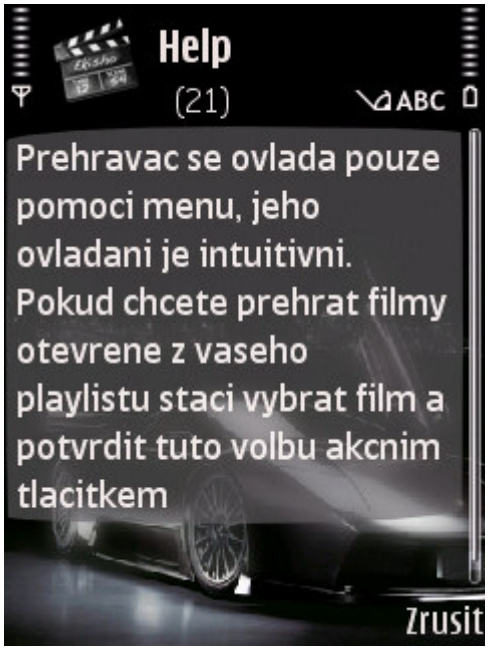
Obr. 13: Souborový manažer

Obr. 14: Otevření streamu

Media Info, kde se nám zobrazí podporované protokoly a typy médií telefonu (viz. Obr. 15). Tyto informace se v závislosti na mobilním telefonu liší, novější typy telefonů podporují velké množství mobilních video formátů a také spoustu přenosových protokolů jako je rtsp nebo http. Po volbě Help se zobrazí krátká nápověda (viz. Obr. 16). Další obrázek již ukazuje připravené video a odkaz na toto video (viz. Obr. 17). Tento prvek jsem zařadil do aplikace kvůli demonstrativnímu účelu, aby ukázal celou cestu k souboru. A také aby si uživatel nemyslel, že aplikace „zamrznula“ nebo něco podobného, neboť příprava videa na přehrávání nějakou dobu trvá, tudíž nejde vidět žádná změna.

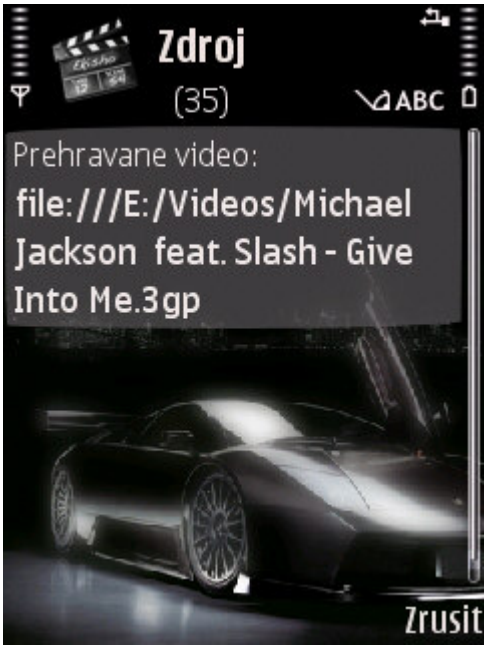


Obr. 15: Multimediální typy



Obr. 16: Krátká nápověda

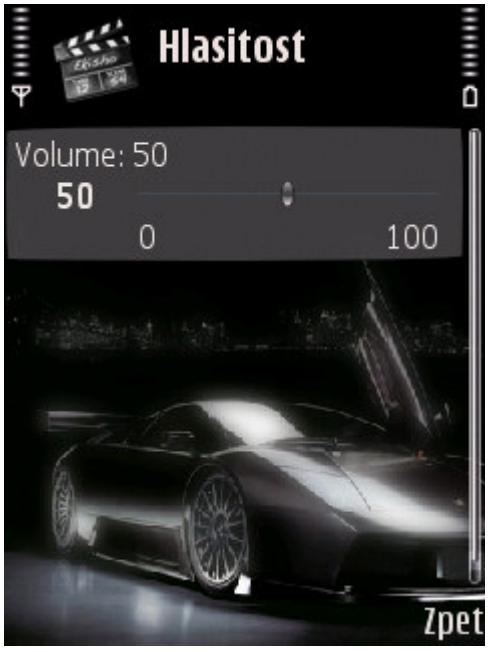
Obr. 18 demonstruje již přehrávané video, které je umístěno na grafickém prvku Canvas z důvodu lepší manipulace s videem a pozdějším úpravám. Rozšířené menu o prvky pro ovládání přehrávaného média je vyobrazeno na Obr. 19. Jsou zde položky Pauza, pro pozastavení přehrávání, Play, pro opětovné spuštění přehrávání média, položka Stop, která slouží pro zastavení přehrávání. Další položkou je položka Mute, která ztiší hlasitost přehrávaného zvuku. Volbou Hlasitost se dostaneme k ovládání hlasitosti pomocí posuvníku (viz. Obr. 20).Ostatní položky jsou stejné jako v základním menu i se stejnou funkcí.



Obr. 17: Odkaz na video

Obr. 18: Přehrávané video

Obr. 19: Rozšířené menu



Obr. 20: Ovládání hlasitosti

Poslední obrazovka, která se nám může v této aplikaci zobrazit, je pokud otevřeme místo multimediálního souboru soubor s příponou .csv, který slouží jako playlist. Poté se nám zobrazí obrazovka, kde jsou jednotlivé názvy filmů, které jsme si zakoupili pomocí webového rozhraní (viz. Obr. 21). Po vybrání filmu a potvrzení akční klávesou se nám spustí přehrávání ze vzdáleného serveru. Musíme být trpěliví, neboť přehrávač musí navázat spojení se serverem a poté ještě ukládá datový tok do vyrovnávací paměti. Aplikace se může zdát po tuto dobu nečinná.

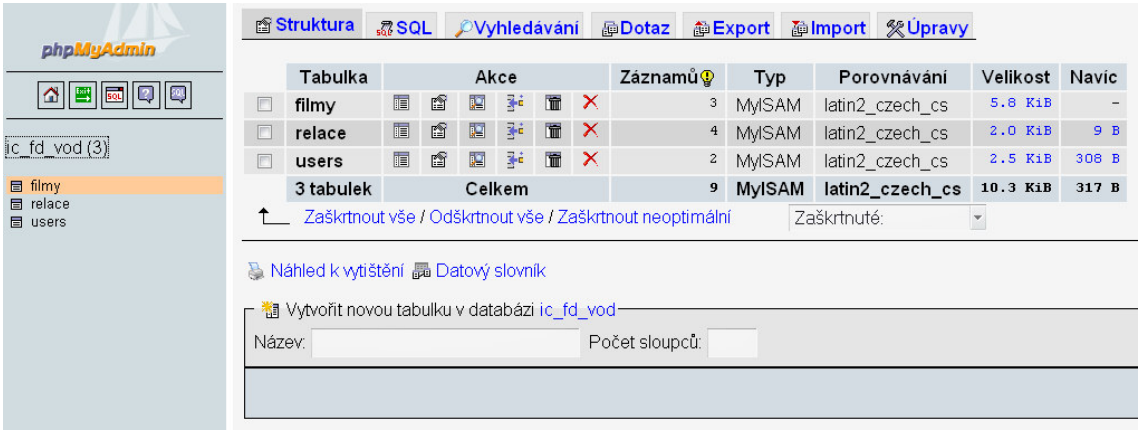
Obr. 21: Playlist aplikace

6.3. *Vypracování serverové části*

Serverová část zahrnuje několik technologií, které jsou vzájemně propojeny tak, aby uživateli poskytovali co nejpohodlnější ovládání celého systému.

Základní část tvoří streamovací server, od společnosti Apple, Darwin Sreaming Server. Tento server jsem zvolil pro jeho vhodné vlastnosti pro toto zadání. Dokáže streamovat přes protokoly RTSP a RTP. Podporuje nativně streamování mnoha formátů, hlavně formát pro přehrávání v mobilních zařízeních 3gp. S jinými servery (VLC Media Player) byly problémy s překódováním videa do formátu, který by byly mobilní zařízení schopny přehrát. Celá konfigurace tohoto serveru probíhá pomocí příkazového řádku, nebo je tu možnost použití webového rozhraní (viz Obr. 9), které je uživatelsky přívětivější. Na serveru musí být ale nainstalován jak streamovací server tak navíc ActivePerl, který provádí parsování ovládacích skriptů streamovacího serveru, které jsou otevírány pomocí webového prohlížeče.

Další součástí serverové části je databáze. Jako nejvhodnější databázi jsem vybral MySQL, která je volně šiřitelná a také si nejlépe rozumí se skriptovacím jazykem PHP. Není sice vhodná pro rozsáhlé databázové systémy, ale pro účely dané touto prací plně vyhovuje. Administraci databáze lze provádět třemi způsoby. První asi nejméně náročný je pomocí phpMyAdmina, který je vytvořen přímo pro ovládání databáze MySQL (viz. Obr. 22). Dalším možným způsobem je ovládání pomocí příkazů, zadávaných do příkazové řádky.

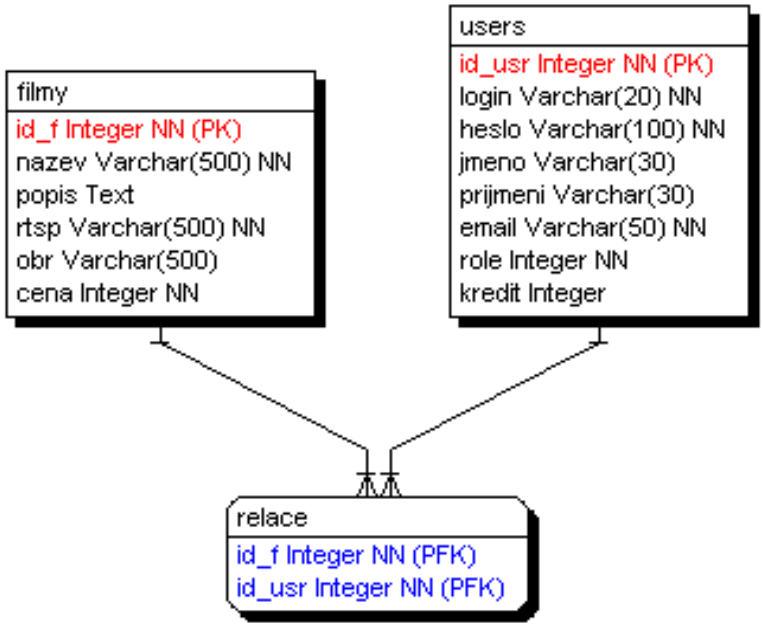


Obr. 22: Administrace MySQL databáze

Nejméně pohodlný a časově náročný způsob ovládání, je ovládání přes ovládací skripty. Tyto skripty si musíme sami vytvořit, například v PHP a pak jimi můžeme databázi ovládat. V práci jsem používal výhod phpMyAdmina pro administraci celé databáze.

Vyčítání, ukládání a modifikace dat v databázi je prováděno pomocí php skriptů, které bylo nutné naprogramovat. V této databázi jsou vytvořeny tři tabulky (viz Obr. 23). První tabulka obsahuje informace o nabízených filmech, název a základní informace o filmu, a rtsp odkaz na tento film. Druhá tabulka obsahuje základní informace o zaregistrovaných uživatelích a poslední informace o vztazích (relacích) mezi uživateli a filmy. Tato relace je typu M:N, kde jeden uživatel může mít zakoupeno více filmů a zároveň jeden film může patřit několika uživatelům.

Pro vyčítání a zobrazování informací z databáze je použit skriptovací jazyk PHP v kombinaci se značkovacím jazykem XHTML ve verzi 1.0 Strict. Vzhled stránek je vytvořen za pomocí kaskádových stylů, které jsou vhodné používat na oddělení obsahu od vzhledu webových stránek. Základem je Apache HTTP Server s modulem pro podporu PHP. Tento server zpracovává požadavky uživatelů, kladené pomocí php skriptů, a zpět k uživateli je posílá jako čisté XHTML stránky, které se zobrazují v klientském prohlížeči webových stránek (viz Obr. 10). Výhodou tohoto řešení je, že je multiplatformní, a tedy nezáleží na operačním systému, který kdo používá. Stačí mít nainstalovaný prohlížeč webových stránek a není třeba dalších speciálních programů na klientské straně. Navržené řešení a zpracování do podoby, kterou vidí samotný uživatel po zadání příslušné adresy do prohlížeče můžeme vidět na Obr. 24.

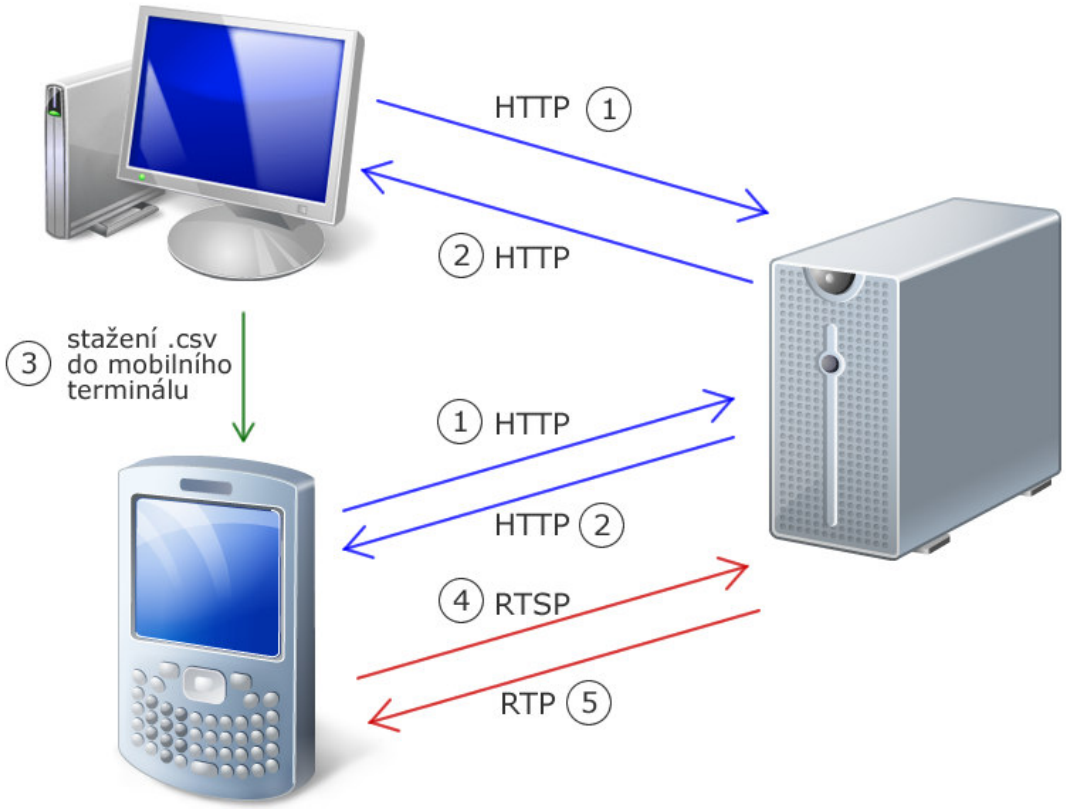


Obr. 23: Struktura databáze

Obr. 24: Úvodní obrazovka webového rozhraní databáze

6.4. Funkčnost celého systému

Celý systém je vytvořen tak, aby byl co nejpřehlednější a uživatelsky přívětivý. Důraz je kladen na ovladatelnost a funkčnost celého systému. Jak je řečeno výše, systém se skládá ze dvou částí, které spolu komunikují. Klientská část sestávající ze samotného přehrávacího klienta pro mobilní zařízení naprogramovaného v jazyce Java ME. Druhou částí je serverová část obsahující streamovací server a webovou aplikaci, díky níž se může uživatel zaregistrovat do systému a po uhrazení poplatku za jednotlivý film si může tento film prohlédnout na mobilním zařízení online. Tedy bez nutnosti mít film uložen na lokálním médiu zařízení. Tím šetří i paměťovým místem na zařízení pro jiné využití. Jedinou nevýhodou vyplývající z tohoto řešení je, že pokud uživatel není v dostupnosti signálu od operátora nemůže službu využívat. To neplatí, pokud se připojuje k internetu jinak než pomocí operátora. Například pokud má zařízení možnost připojení pomocí jiné technologie, která je v danou chvíli k dispozici (např. WiFi). Na Obr. 25 je vidět jak celý systém funguje. Jednotlivé kroky komunikace jsou popsány níže.



Obr. 25: Schéma celého systému

Jednotlivé kroky, které jsou možné v systému provádět:

1. Kontakt serveru pomocí http protokolu a příslušné URL. Pomocí vytvořených skriptů je možné se serverem interaktivně komunikovat a posílat mu jednotlivé požadavky. Základní obrazovka systému po zadání správné adresy je vyobrazena na Obr. 24: Úvodní obrazovka webového rozhraní databáze. Tuto komunikaci můžeme vést z osobního počítače nebo také přímo z mobilního zařízení, pokud nám to toto zařízení poskytuje. Pozn. Nejznámější webový prohlížeč pro mobilní telefony je Opera Mini, která tuto komunikaci bez potíží zvládá a požadované stránky zobrazuje korektně. Navíc je napsána též v jazyce Java ME.
2. Na tyto požadavky server odpovídá opět pomocí protokolu http. Posílá zpět do prohlížeče XHTML stránky, které jsou zobrazitelné ve všech běžných webových prohlížečích (např. Internet Explorer, Firefox, Opera). Během této komunikace je možné se zaregistrovat do systému (viz. Obr. 24, položka menu Registrace), prohlížet si databázi nabízených filmů (viz. Obr. 26), zakoupit si film, s tím, že se nám odečte udávaná částka z našeho kreditu (tlačítko s košíkem na Obr. 26) a poté si vygenerovat soubor (nalezneme v administraci účtů na Obr. 27, položka

Vygeneruj soubor), který si podle připojení uložíme buď do počítače nebo přímo do mobilního zařízení. Tento soubor nám později bude sloužit jako playlist při přehrávání souborů ze serveru.

3. Připojení mobilního terminálu k osobnímu počítači, za pomoci kabelu nebo bezdrátově (BlueTooth, infraport, WiFi), a zkopírování vygenerovaného souboru na paměťové médium zařízení. Tento krok můžeme vynechat pokud se připojujeme z mobilního terminálu, kde uložíme soubor přímo na médium v zařízení.
4. Otevření playlistu z daného umístění v zařízení. K tomu nám slouží položka Otevřít v klientské aplikaci (viz. Obr. 12). Poté v souborovém manageru (viz. Obr. 13) vybereme námi stažený soubor ze serveru. Tento soubor má tvar login.csv, kde login je vaše přihlašovací jméno. Po otevření souboru se nám již zobrazí playlist, názvy filmů, které máme zakoupené (viz. Obr. 21). Po vybrání příslušného filmu ze seznamu, si vybereme příslušný řádek seznamu a potvrdíme akčním tlačítkem. Ten se nám po chvíli začne přehrávat (pozor na zpoždění dané nutností navázání spojení a uložení části vybraného média do vyrovnávací paměti, aplikace se nám může jevit jako nefunkční – „zamrzlá“). Aplikace pomocí RTSP protokolu kontaktuje streamovací server na dané adrese, která je uložena v rámci playlistu, a dohodnou si spolu potřebné údaje pro komunikaci. Přehrávání již může začít. Na obrazovce mobilního zařízení se zobrazí již přehrávané médium a po stisku tlačítka Volby se nám zobrazí rozšířené menu (viz. Obr. 19), kde můžeme médium ovládat.
5. Posledním krokem je již přehrávání vybraného multimediálního obsahu, který byl domluven za pomoci protokolu RTSP v předchozím kroku. Tento tok dat je přenášen sítí pomocí protokolu RTP. Po výběru volby Stop v rozšířeném menu aplikace, se celé spojení uzavře a přehrávání se zastaví. Opětovné přehrávání spustíme opětovným výběrem položky Otevřít v menu, nebo pomocí položky Otevřít stream, kdy jako adresu zadáme přímo rtsp odkaz námi zvoleného média.

Celé toto řešení bylo otestováno lokálně. Testovací sestava obsahovala osobní počítač s nainstalovaným streamovacím serverem DSS, Apache HTTP serverem a databází MySQL. Aplikace byla nainstalována na mobilním telefonu značky Nokia model E51, který má integrovanou technologii WiFi, pomocí které byly uskutečňovány datové přenosy. Dalším prvkem při testování byl přístupový bod od firmy Linksys. Po

fázi testování byla webová aplikace s databází přemístěna na internet, kde je dostupná pod adresou www.fd-vod.ic.cz. Na této adrese jsou dostupné informace o tom jak celé řešení uživatelsky zprovoznit a také je zde dostupná aplikace pro mobilní zařízení. Stačí si tedy tuto aplikaci stáhnout, nainstalovat do mobilního zařízení a poté již podle postupu výše sledovat video na vyžádání. Tento zveřejněný systém využívá pro stremování videa z portálu YouTube pro mobilní zařízení, který je dostupný na adrese www.m.youtube.com.

Obr. 26: Databáze nabízených filmů

Obr. 27: Administrace účtu

Závěr

Platforma JavaME je jednou z dnes nejrozšířenějších platforem pro tvorbu aplikací pro mobilní zařízení. Jednotlivá API konfigurací a profilů jsou relativně dobře srozumitelná a dobře se s nimi pracuje. Při tvorbě aplikace se vyskytlo několik problémů, které však byly způsobeny nedokonalou znalostí platformy. Jediný podstatný problém kompletní aplikace shledávám v řešení přístupu k souborovému systému, kde jsou výhody a nevýhody diskutovány v kapitole 6.2. Při testování programu na jednotlivých emulátorech a mobilních zařízeních jsem neshledal žádné nestandardní chování. Při testování se mi také potvrdilo, že se aplikace chovala různě na různých zařízeních díky volnosti implementace jednotlivých funkcí různými výrobci. Aplikace není podepsaná, tudíž při přístupu k souborovému systému a síti vyžaduje potvrzování práv k přístupu uživatelem.

Na serverové straně byl nejdůležitější návrh a zpracování databáze nahrávek. S navrženým řešením nebyly žádné problémy při testování. Nejsložitější operací byl výběr vhodného streamovacího serveru, který by dokázal streamovat pomocí protokolu RTSP a také který by dokázal upravit streamované video do formátu vhodného pro přenos sítí. Testem prošlo několik serverů, ale nakonec jako nejvhodnější varianta byl zvolen server od firmy Apple Darwin Sreaming Server, který splňoval veškeré požadavky kladené na celý systém.

Test celého systému proběhl úspěšně, jeho výsledky jsou k dispozici v kapitole 6.4. Konečné řešení bylo zveřejněno na webové adrese www.fd-vod.ic.cz. U toho zveřejněného systému je využíváno služeb části YouTube, které jsou uzpůsobeny pro vysílání videa pro mobilní zařízení. To s sebou nese mnoho výhod, včetně toho, že není třeba se starat o práva doručovaného obsahu.

Literatura

[1] GOYAL, Vikram. *Pro Java ME MMAPI : Mobile Media API for Java Micro Edition*. Apress, 2006. 286 s. ISBN 1-59059-639-0.

[2] Brief Introduction to the Mobile Media API. *Forum Nokia* [online]. 2003 [cit. 2008-12-10]. Dostupný z WWW: www.forum.nokia.com.

[3] Video And Streaming In Nokia Devices. *Forum Nokia* [online]. 2005 [cit. 2008-12-10]. Dostupný z WWW: www.forum.nokia.com.

[4] GOYAL, Vikram. *J2ME Tutorial, Part 4: Multimedia and MIDP 2.0* [online]. 1995-2008 , 09/27/2005 [cit. 2008-12-10]. Dostupný z WWW: http://today.java.net/pub/a/today/2005/09/27/j2me4.html.

[5] *Java ME* [online]. 2008 , 13. 5. 2008 [cit. 2008-12-10]. Dostupný z WWW: http://kore.fi.muni.cz:5080/wiki/index.php/Java_ME.

[6] QUASAY, Mahmoud. *The J2ME Mobile Media API* [online]. 1994-2008 , June 2003 [cit. 2008-12-10]. Dostupný z WWW: http://developers.sun.com/mobility/midp/articles/mmapioverview.

[7] *RTP, Real-Time Transport Protocol* [online]. c1998-2009 [cit. 2009-05-15]. Dostupný z WWW: http://www.networksorcery.com/enp/protocol/rtp.htm.

[8] *Real-Time Transport Protocol* [online]. [2009] , 18 May 2009 [cit. 2009-05-15]. Dostupný z WWW: http://en.wikipedia.org/wiki/Real-time_Transport_Protocol.

[9] *Multimedia over the Internet* [online]. c2000 [cit. 2009-05-15]. Dostupný z WWW: https://prof.hti.bfh.ch/myf1/www/projects/polyphem/www/documents/projectwork-techreport-mediainternet.html.

[10] *Real Time Streaming Protocol* [online]. [2009] , 8 May 2009 [cit. 2009-05-15]. Dostupný z WWW: http://en.wikipedia.org/wiki/Real_Time_Streaming_Protocol.

[11] *Streaming Server* [online]. c2009 [cit. 2009-05-16]. Dostupný z WWW: http://developer.apple.com/opensource/server/streaming/index.html.

[12] *Apache HTTP Server* [online]. [2009] , 17 May 2009 [cit. 2009-05-16]. Dostupný z WWW: http://en.wikipedia.org/wiki/Apache_HTTP_Server.

[13] Kosek, J.: *PHP-Tvorba interaktivních internetových aplikací*, Praha, Grada Publishing, 1999, ISBN 80-7169-373-1

[14] Staníček, P.: *CSS Kaskádové styly*, Brno, Computer Press, 2003, ISBN 80-7226-872-4

A Obsah přiloženého CD

Mobilni_prehravac_v3

Mobilni_prehravac_v3_jar

Webova_aplikace

Zaloha_Databaze

Text diplomové xobdrz30_diplomova_prace.pdf

Adresář Mobilni_prehravac_v3 obsahuje aplikaci napsanou v Java ME. Adresář je přímo projektem vývojového prostředí Netbeans 6.1, ve kterém je spustitelné a lze projekt spustit. Součástí Netbeans musí být i Mobility Pack.

Druhý adresář Mobilni_prehravac_v3_jar obsahuje instalační balíček .jar, který jde přímo nainstalovat do zařízení podporující JAVA aplikace.

Adresář Webova_aplikace obsahuje php skripty pro ovládání databáze a zobrazování výsledků ve webovém prohlížeči.

Poslední adresář Zaloha_databaze obsahuje zálohu celé databáze, ve formátu .sql, kterou lze naimportovat do jakékoliv jiné databáze.

Dále CD obsahuje text diplomové práce ve formátu .pdf