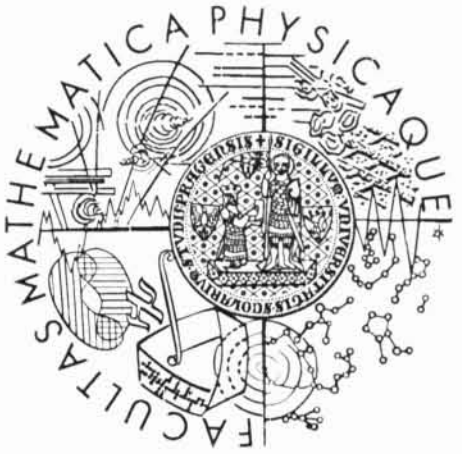
Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

BAKALÁŘSKÁ PRÁCE



Jiří Bláha

Katalog a distribuce videosouborů

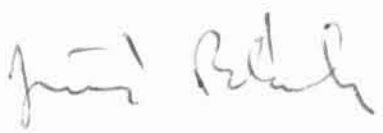
Katedra softwarového inženýrství

Vedoucí bakalářské práce: RNDr. Antonín Kosík
Studijní program: informatika, obecná informatika

2006

Prohlašuji, že jsem svou bakalářskou práci napsal samostatně a výhradně s použitím citovaných pramenů. Souhlasím se zapůjčováním práce a jejím zveřejňováním.

V Praze dne 11. srpna 2006 Jiří Bláha



Obsah

Název práce: Katalog a distribuce videosouborů
Autor: Jiří Bláha
Katedra (ústav): Katedra softwarového inženýrství
Vedoucí bakalářské práce: RNDr. Antonín Kosík
e-mail vedoucího: Antonin.Kosik@mff.cuni.cz

Abstrakt: Cílem práce je prozkoumat různé způsoby distribuce videosouborů na Internetu a navrhnout a popsat webovou aplikaci pro jejich snadnou katalogizaci a distribuci. Obecně pro distribuci větších objemů dat jsou vhodné P2P sítě, práce se věnuje zejména síti BitTorrent. Přímo pro distribuci videa lze použít stále oblíbenější podcasting, jehož implementace je velmi jednoduchá. Klasickou metodou je také streamování videa na požádání. Velmi oblíbené je dnes i Flash video. Úkolem aplikace je také katalogizace videosouborů. Katalog je implementován jako databázová aplikace psaná v jazyku PHP nad open source SQL databází, je dostatečně univerzální a nastavitelný.

Klíčová slova: video, distribuce, katalog, bittorrent, podcasting

Title: Distribution and catalogue of video files
Author: Jiří Bláha
Department: Departement of Software Engineering
Supervisor: RNDr. Antonín Kosík
Supervisor's e-mail address: Antonin.Kosik@mff.cuni.cz

Abstract: The aim of this work is to look into various ways of distributing video files over Internet and to design and describe web application for their easy cataloguing and distribution. Generaly, the P2P networks are suited for distribution of bigger amounts of data, this work deals especially with the BitTorrent network. For distribution of video data, we can use podcasting, which is becoming more and more popular, and which is quite easy to implement. Rather classical method is streaming on demand. Nowadays, the Flash video is quite popular too. The task of the application is to catalogue the video files as well. The catalogue is implemented as a database application written in the PHP language over an open source SQL database, it is universal and configurable enough.

Keywords: video, distribution, catalogue, bittorrent, podcasting

Kapitola 1

Úvod

Cílem práce je popsat běžně používané technologie pro šíření videosouborů na Internetu a navrhnout jednoduché řešení pro jejich katalogizaci a distribuci prostřednictvím WWW aplikace postavené na technologiích XHTML/CSS/PHP/SQL. Řešení by přitom mělo být značně univerzální a využívat co největší spektrum v dnešní době rozšířených technologií používaných jednak pro šíření větších objemů dat obecně, především však technologií specifických pro distribuci multimédií. Zohledněna by přitom měla být snadná instalace a možnost nasazení na běžných hostingových serverech bez nutnosti běhu specializovaných služeb.

Univerzálnost aplikace by měla spočívat především ve vysoké míře nastavitelnosti (použití jen některých skriptů, volba grafických vzhledů, volba jazyka, databázová abstrakce) a částečně v rozšiřitelnosti (doplňování dalších technik pro distribuci souborů, vytváření vlastních grafických vzhledů pomocí CSS, jednoduchá lokalizace do jiného jazyka). Na druhou stranu je třeba, aby šířka možností nastavení nebyla na úkor jednoduchosti instalace a používání.

Kapitola 2

Distribuce videosouborů

Pro distribuci videa můžeme použít obecně jak metody pro šíření větších objemů dat obecně, tak metody specifické pro šíření dat multimediálních. Pro účely práce bude rozhodujícím kritériem pro posouzení jejich použitelnosti především snadnost instalace a použitelnost v rámci webové aplikace.

V této kapitole shrnu možné způsoby šíření vidoobsahu a naznačím způsob, jakým budou v aplikaci implementovány.

2.1 Klasický download

Základní variantou je klasický download souboru. Rozšířené jsou dvě varianty - stažení přes HTTP, nebo FTP protokol. Tato varianta je samozřejmě implementačně nejjednodušší a uvádím ji jen v zájmu úplnosti.

2.2 P2P Sítě, BitTorrent

Výměnné P2P sítě (nemám zde namysli peer-to-peer sítě v obecném slova smyslu, ale P2P sítě ve smyslu aplikačních protokolů určených pro výměnu dat mezi uživateli) jsou, i přes více či méně úspěšné zásahy různých organizací starajících se o dodržování autorských práv, stále více oblíbené. Z nejrozšířenějších jmenujme sítě Gnutella, eDonkey2000, FastTrack, Direct Connect a BitTorrent. Výhoda většiny takových sítí (pomineme-li nelegální šíření autorsky chráněných dat) je ta, že o šíření konkrétního souboru se obvykle dělí více uzlů a síťový provoz je tak, v porovnání například s klasickým downloadem přes HTTP protokol, rovnoměrněji rozložen. Toho může využít

uzel, který nabízí nějaký obsah, a často také využívá, například linuxové distribuce je běžně možné stahovat prostřednictvím sítě BitTorrent. Právě protokol BitTorrent je svou koncepcí pro tyto účely velmi vhodný.

BitTorrent Network je oblíbená P2P síť, jejíž obrovskou výhodou je velká míra decentralizace. Abychom mohli soubory distribuovat přes síť BitTorrent, nepotřebujeme, aby kvůli tomu na našem serveru běžela speciální služba. Abych mohl srozumitelně popsat, jak síť BitTorrent funguje, vysvětlím nejprve některé základní pojmy:

- **Klient** je program, pomocí kterého uživatel stahuje soubory od jiných uživatelů sítě BitTorrent. Pomocí klientu uživatel zároveň poskytuje své soubory (i ty, které zrovna stahuje) ostatním. Pokud jsou dva klienty spojené, veškerá jejich komunikace probíhá jen mezi nimi, bez dalšího prostředníka.

- **Tracker** je (v zásadě velmi jednoduchá) aplikace, která pomocí HTTP protokolu zprostředkovává spojení mezi jednotlivými klienty. Zároveň o nich udržuje – vesměs statistické – informace. Informace o tom, kdo disponuje kterým souborem, kolik už má kdo z kterého souboru staženo, jakou rychlostí kdo stahuje apod. Klient v pravidelných intervalech kontaktuje tracker, předává mu informace o sobě a na oplátku dostává informace o ostatních klientech. Tracker vůbec „nevidí" vlastní soubory, které si mezi sebou klienty vyměňují. Jde zpravidla o relativně jednoduchý skript (například PHP skript, jako v našem případě), umístěný na nějakém veřejně přístupném webovém serveru. Trackerů existuje na Internetu velké množství, každý si může vytvořit a provozovat svůj vlastní. Poslední dobou se objevují i „uzavřené", tzv. *privátní* trackery, které vyžadují k jejich používání registraci a můžou definovat i další pravidla (jako minimální poměr upload/download).

- **Torrent** je malý binární soubor s příponou `.torrent`, který uchovává informativní údaje o vlastním sdíleném souboru (či souborech) – tzv. *metainformace*. Obsahuje důležité prvotní informace, které jednak můžou posloužit jako vodítko uživateli, hlavně však samotném klientu. Ten se z něj dozví především URL adresu trackeru. Další údaje obsažené v torrentu jsou jména jednotlivých souborů, jejich velikosti, velikost jednoho dílu (soubory jsou logicky dělené na díly a klient je stahuje nezávisle na sobě) a *SHA-1 haše* pro každí díl, které slouží pro kontrolu jejich intergrity po stažení. Krom toho obsahuje torrent

další nepovinné položky, jako komentář, jméno autora nebo datum vydání. Jak již bylo naznačeno, torrent může „slučovat" i větší množství souborů, a to včetně adresářové struktury.

- **Seeder** je klient, který disponuje kompletně staženým souborem a už ho jen nabízí ke stažení ostatním.

- **Peer** je klient, který soubor ještě kompletně stažený nemá, tento termín se ale často používá pro označení libovolného klienta.

Distribuce souborů pomocí sítě BitTorrent probíhá následovně. Uživatel, který chce sdílet nejaká data, nejprve z těchto dat ve svém klientu vytvoří torrent a uvede adresu trackeru. Stává se tak prvním seederem. Torrent pak libovolným způsobem (nejčastěji umístěním na web) dá k dispozici dalším uživatelům. Pokud jej někdo další otevře ve svém klientu, ten kontaktuje uvedený tracker, předá mu své „kontaktní" údaje (IP adresu, číslo portu, na kterém poslouchá a tzv. *peer_id*, které si sám vygeneruje) a dozvídá se od něj, od koho může soubor(y) stahovat (tj. IP adresy a čísla portů dalších klientů). V pravidelných intervalech (zpravidla v řádu desítek minut; délku intervalu vrací tracker klientovi v odpovědi) se tato procedura opakuje s tím, že klient průběžně „hlásí" svůj pokrok. Komunikace mezi klientem a trackerem, jak již bylo řečeno, probíhá prostřednictvím protokolu HTTP. Klient předává údaje metodou GET a dostává odpověď ve formátu text/plain. Přesnou specifikaci lze nalézt na [1].

Naše aplikace má za úkol distribuovat videoobsah uložený na našem serveru, proto bude prvním seederem. Naprogramování vlastního trackeru je relativně jednoduchá záležitost (navíc existuje několik povedených implementací s otevřeným kódem, které je možné použít). Potíž je ale s klientem, který musí, chceme-li soubori pomocí sítě BitTorrent distribuovat, běžet neustále. Navíc jde už o netriviální program, jehož napsání je již nad rámec této práce. Pokusím se tedy použít nějaké hotové řešení, které by bylo pokud možno platformně nezávislé. Jako vhodný se jeví program **Azureus**, který je psán v jayzku Java a poběží tedy na libovolné platformě, na které již běží *Java Virtual Machine*.

2.3 Podcasting, Vodcasting

Podcasting je poměrně nová metoda šíření souborů „vynalezená" v roce 2004. Je určena především pro distribuci audio a videosouborů. Je založena

na jednoduché myšlence využití RSS pro šíření binárních dat, používá se proto také termín Audio/Video RSS. Jediný rozdíl proti distribuci „klasickým downloadem" přes HTTP, je v podstatě ten, že RSS klient (v případě podcastingu také nazývaný *podcatcher* nebo *podcast receiver*) soubory stahuje automaticky do klientova počítače.

Tento specializovaný program bývá v zájmu uživatelského pohodlí přímo spojen s mediálním přehrávačem. Původním využitím podcastingu bylo automatické stahování audiosouborů do hardwaravých přehrávačů iPod firmy Apple, pro svůj jednoduchý princip se však podcasting již poměrně rozšířil a bývá využíván například pro automatické stahování pořadů z internetových rádií nebo televizních seriálů (uživatel je pak může poslouchat či sledovat nezávisle na tom, kdy byly skutečně vysílány).

Pro podcasting videosouborů se později začal používat speciální termín vodcasting, technologie je však stejná. Je také tendence odlišovat podcasting/vodcasting ve smyslu pravidelné distribuce obsahu (například radiových pořadů či televizních seriálů) od takzvaných audio/video deníčků (audio blog, video blog, vblog), kde je naopak obsah uvolňován spíše nepravidelně. Jde vlastně jen o různá pojmenování identické technologie a pro moji práci však nejsou tyto drobné odlišnosti v terminologii podstatné.

Výsledná aplikace by měla být schopná distribuovat odkazy na videosoubory pomocí RSS kanálů. Bude generovat XML dokumenty, který budou srozumitelné pro Podcastové klienty. Takový klient ale bývá, jak již bylo řečeno, obvykle přímo spojen s multimediálním přehrávačem, který nemusí nutně umět přehrát všechny nabízené videoformáty. Proto bude možné nastavit, aby aplikace nabízela několik různých RSS kanálů, například právě pro různé formáty videosouborů.

2.4 Streaming, Video On Demand

Některé videoformáty (asi by bylo lepší napsat kontejnery, protože v praxi tolik nezáleží na použitém audio či videokodeku) mohou být přehrávány už v době, kdy je klient stahuje. Termín streaming se používá spíše ve spojení s živým vysíláním, naopak termín **video on demand** (česky doslova *video na požádání*) použijeme obvykle při distribuci audiovizuáního materiálu uloženého v souboru na serveru.

Nejznámějšímy, a vlastně jedinými dostatečně rozšířenými formáty, které umožňují streaming, jsou **Windows Media** společnosti Microsoft, **RealAudio** resp. **RealVideo** společnosti Real Networks nebo **QuickTime** firmy

Apple Computers. Nesmíme určitě zapomenout zmínit opensource kontejner **Ogg** komunity Xiph.org, který plně plně podporuje streaming. Využívá se obvykle ve spojení s audiokodekem **Vorbis** a videokodekem **Theora**, které jsou produktem stejné opensource kommunity. Jeho rozšíření brání především kvalita videokodeku, která je ve srovnání se zmíněnými komerčními produkty zatím poměrně nízká.

Video může být streamováno v zásadě dvěma způsoby. Tím prvním, jednodušším, ale pro daný účel ne zcela vhodným je opět využití HTTP protokolu. Hlavní, a vlastně i jedinou, výhodou je jednoduché použití. Tato varianta se do downloadu liší prakticky jen tím, že uživatel může začít video sledovat, ještě než ho má celé stažené.

Druhou možností je provoz specializovaného streamovacího serveru (Streaming server). To má mnoho výhod – v první řadě lze využít zvláštní aplikační protokol, který je pro tento účel dělaný. Ten pak zpravidla poběží nad transportním protokolem UDP, který je nespolehlivý, a výhody s tím spojené šíření multimediálních dat nahrávají. Klient se serverem mohou komunikovat úplně na jiné kvalitativní úrovni, než je tomu při komunikaci přes HTTP protokol – server se může klientovi přizpůsobovat, měnit dynamicky datový tok a podobně. Dále je možné využít multicastu, který výrazně, až drasticky, snižuje síťovou zátěž tím, že umožňuje přehrávání jednoho streamu několika uživatelům současně. Multicast má význam samozřejmě jen při „živém" streamingu. Jedinou nevýhodou (vzhledem k vytyčeným cílům práce však poměrně významnou) je právě nutnost provozování zvláštní služby na serveru. Vzhledem k tomu, že výsledná aplikace by měla být použitelná na běžném webhostingu, není tento způsob distribuce videosouborů vhodný.

2.5 Flash Video

Protože se stal Macromedia Flash (dnes již Adobe Flash) de facto webovou technologií, jeví se jeho využití pro sledování videa na webu jako ideální řešení. Už od verze 6 dokáže Flash Player přehrávat videa komprimovaná variantou kodeku H.263. Tato videa jsou šířena v souborech s příponou FLV (zkratka z Flash Video), které mohou být uzavřena ve známých souborech SWF. Poslední dobou se tato forma šíření videa stala velmi populární, hlavně kvůli velké oblíbenosti služeb pro sdílení videa¨Google Video a YouTube, které tento způsob obě využívají. Videosoubory ve formátu FLV není složité vyrobit, existují komerční i volně šiřitelné aplikace pro převod z jiných formátů.

V praxi je součástí stránky flashový prvek (soubor ve formátu SWF), který funguje jako přehrávač souboru FLV. Video samotné je uloženo zvlášť – není zakomponované přímo v SWF – a stahuje se až dodatečně, například po klepnutí na tlačítko play v přehrávači.

Kapitola 3

Implementace

V této kapitole podrobněji popíši, jak výsledná aplikace funguje. Nejprve se stručně zmíním o použitých technologiích, poté popíši, jak je aplikace rozdělená do skriptů a jaká je struktura databáze, zmíním některé zajímavé postupy a rozeberu řešení multijazykovosti, databázové abstrakce a implementaci jednotlivých technologií pro šíření videosouborů.

3.1 Použité webové technologie

Aplikace je psána kombinací „webových jazyků“ XHTML/CSS a hypertextového preprocesoru PHP ve verzi 4 (veškerý kód by ale měl být funkční i v PHP5). Co se týče PHP kódu, předpokládám nastavení `register_globals = off` v `php.ini`, tj. pro práci s „externími“ proměnnými používám asociativní pole `$_GET`, `$_POST`, `$_REQUEST`, `$_SESSION` a `$_SERVER`. Dále, kde je to vhodné, používám zjednodušené objektově orientované programování, které jazyk PHP nabízí.

Jde o databázovou aplikaci, není však jednoznačně dáno, jaký databázový stroj se použije. Používám jazyk SQL ve variantě, kterou používá databáze MySQL verze 4, tedy bez procedurálního rozšíření (PL/SQL) a bez vnořených dotazů, tzv. *subqueries*. Z toho často plyne nutnost řešit operace s daty, které by šly řešit už na úrovni databázového stroje, až na úrovni aplikační. Právě MySQL server verze 4 je ale stále jediný databázový stroj na velké části hostingových služeb.

Samotný (X)HTML kód je psán s ohledem na moderní trendy oddělování obsahu a formy a je tedy kompletně *skinovatelný* pomocí CSS stylů.

3.2 Struktura aplikace

Generování stránky Základem aplikace je skript `index.php`, do kterého jsou pak vkládány další skripty a dokumenty v závislosti na parametru předaném metodou `GET` nebo `POST`. Ty potom odpovídají jednotlivým sekcím stránek. Do souboru `index.php` jsou však v první řadě *includovány* další, především různé konfigurační skripty, konkrétně jsou to soubory (v pořadí, v jakém se do skriptu vkládají):

- `fce.ini.php`, kde jsou definovány funkce používané na různých místech celé aplikace
- `db.ini.php`, kde jsou definovány konstanty pro připojení k databázi
- `config.ini.php`, ve kterém správce aplikace definuje další konstanty a nastaví různé proměnné, které ovlivňují vzhled i funkčnost aplikace
- soubor, ve kterém jsou definovány řetězcové konstanty jazyka, ve kterém aplikace komunikuje. Jeho název je tvaru *`jazyk`*`.lang.php`
- `kontrola.php`, který kontroluje oprávněnost uživatele používat aplikaci
- a konečně `var.ini.php` s definicemi dalších – interních – proměnných aplikace. Ten je vkládán až poslední, neboť již vyžaduje zavedené jazykové definice.

Dále se v tomto hlavním souboru zde spustí *session*, nastaví se některé globální proměnné a provede se připojení k databázi.

Protože v některých sekcích je potřeba provést některé akce ještě před započetím odesílání textového výstupu, tj. ještě před odesláním HTTP hlaviček, odpovídají těmto sekcím vždy dva PHP soubory. První, který má tvar *`sekce`*`_pre.php` se vkládá těsně před započetím generování HTML výstupu. Provádějí se v něm vesměs aktualizace databáze (vkládání a úprava záznamů). Poté dojde zpravidla k přesměrování na jinou stránku pomocí HTTP hlavičky `Location`, proto se tedy tento soubor musí zpracovat před výstupem HTML kódu. Druhý, který se až na příponu `pre` jmenuje stejně se vkládá až uvnitř těla HTML dokumentu.

Adresářová struktura V kořenovém adresáři WWW stromu jsou jen tři soubory. Již zmiňovaný `index.php`, dále soubor `popup.php`, který plní podobnou úlohu jako `index.php`, ale pro vyskakovací *popup* okna a konečně soubor `.htaccess`, který upřesněje nastavení webového serveru Apache (konkrétně v této aplikaci je jeho úkolem jen překlad URL adres). Adresářová struktura je následující (v abecedním pořadí):

`bittorrent` Adresář, ve kterém jsou zdrojové kódy BitTorrentového trackeru a podadresář s uloženými torrenty.

`class` Obsahuje zdrojové PHP soubory s třídami pro práci s databází. Konkrétně jsou to třídy Autori pro práci s tabulkou autorů, Ciselniky pro práci s číselníky klíčových slov, žánrů apod., DB pro práci s různými databázovými stroji na nejnižší úrovni, Filmy pro práci s tabulkou filmů a Uziv, která zapouzdřuje práci s tabulkou uživatelů databáze po řadě v souborech `autori.class.php`, `ciselniky.class.php`, `db.class.php`, `filmy.class.php` a `uziv.class.php`.

`config` V tomto adresáři jsou již zmiňované soubory `db.ini.php` a `config.ini.php`.

`images` Adresář s obrázky. Protože obrázky tvořící *design* webu jsou ukládány jinde v rámci grafických skinů, je v tomto adresáři jen další podadresář `scr` obsahující snímky (*screenshoty*) k jednotlivým filmům či videím

`inc` Obsahuje soubory vkládané do skriptu `index.php`. Vesměs jsou to skripty odpovídající jednotlivým sekcím webu.

`lang` Adresář s definicemi textových řetězců pro jednotlivé jazyky. Řetězce pro češtinu jsou například definovány v souboru `cz.lang.php`.

`popup` Zde jsou uloženy skripty, které se *includují* do souboru `popup.php`, podobně jako se skripty z adresáře `inc` vkládají do `index.php`.

`styles` Místo pro uložení vytvořených „grafických“ podob webu – *skinů*. Dále děleno do podadresářů odpovídajících jednotlivým skinům.

`video` Úložiště samotných videosouborů. Tento adrasář lze v konfiguračním skriptu nadefinovat, takže v principu může být kdekoliv, i zcela mimo strom WWW serveru (například nechceme-li nabízet přímý download souboru přes protokol HTTP).

3.3 Katalog videosouborů

Účelem aplikace neměla být jen prostá distribuce videí, ale především také jejich účinná katalogizace. Výchozím předpokladem byla univerzálnost takového katalogu. Měl by být schopný posloužit jak k nabízení videí ke stažení, rozdělených jen do několika kategoriích (nebo vůbec nedělených), stejně jako katalog studentských prací nebo katalog videí nějaké komunty, kde je žádoucí katalogiz ovat i autory. V neposlední řadě i jako prostý katalog filmů bez nabízení samotných videosouborů, jakási „odlehčená verze imdb.com". V této části popíši databázovou strukturu a fungování katalogu.

Katalog má dvě hlavní, vzájemně provázané, části – samotný katalog videí či filmů (nadále budu v této sekci mluvit o filmech) a evidenci jejich autorů.

Nejprve se stručně podíváme na strukturu databáze Nejdůležitější tabulka filmy má následující základní strukturu (vynechal jsem indexy):

```
CREATE TABLE filmy (
        id_filmy int(11) NOT NULL auto_increment,
        url varchar(200) NOT NULL default '',
        soubor text NOT NULL,
        nazev varchar(200) NOT NULL default '',
        delka int(11) NOT NULL default '0',
        rok int(11) NOT NULL default '0',
        popis text NOT NULL,
        datum int(11) NOT NULL default '0',
        PRIMARY KEY (id_filmy)
);
```

Jediné zajímavé atributy jsou url, kam se ukládá řetězec vytvořený z názvu filmu (odstraněním diakritiky a převedením bílých znaků na pomlčky) použitý v takzvaných *hezkých URL*, dále atribut soubor, který obsahuje cestu k videosouboru v rámci adresáře definovaného jako *Video Root* a konečně atribut datum určující datum zařazení filmu do katalogu.

S tabulkou filmů je úzce svázáno několik dalších tabulek, které souhrně označme jako *číselníky*. Konkrétně jde o tabulky keywords, zanry, jazyky a zeme. Obsahují po dvou atributech - např. id_keywords (primární klíč), nazev v případě tabulky keywords, u ostatních analogicky. S tabulkou filmy jsou ve vztahu $M : N$. O propojení se pak starají po řadě tabulky filmy_keywords,

filmy_zanry, filmy_jazyky a filmy_zeme. Mají logicky jen dva atributy, které tvoří primární klíč (například u tabulky filmy_keywords jsou to atributy id_filmy a id_keywords).

Tabulka autorů už není tak zajímavá, proto zde nebudu její strukturu uvádět. Její atributy (kromě primárního klíče id_autori a důležitých atributů jmeno a prijmeni) jsou v zásadě jen informativní. Tabulky filmy a autori samozřejmě také ve vztahu $M : N$ a jsou propojeny další tabulkou filmy_autori. Ta má zvláštní primární klíč id_filmy_autori (jeden člověk se na filmu mohl podílet ve více rolích), dále cizí klíče id_filmy a id_autori a krom toho atributy id_role – cizí klíč tabulky role (to je vlastně číselník filmových profesí - scénář, režie, hlavní role apod.) –, postava, který reprezentuje jméno postavy v daném filmu a hlavni který určuje, jestli se má autor uvádět v seznamech filmů.

Nakonec se zmíním o tabulce uziv, jejíž záznamy odpovídají uživatelům aplikace. Atributy jsou zde primární klíč id_uziv, dále login, heslo - SHA-1 otisk zadaného hesla, jmeno, prijmeni, email, login_cas - datum posledního přihlášení do aplikace. Vztahem $1 : N$ je s ní spojena tabulka uziv_prava, ve které jsou definována práva uživatelů na akce uvnitř aplikace.

Ze struktury databáze je spousta věcí jasných. Ke každému filmu máme možnost přiřadit libovolný počet klíčových slov, jazyků a zemí vzniku, můžeme jej zařadit do libovolného počtu žánrů. Dále k filmu lze přiřadit autory z tabulky autori a u každého určit úlohu, kterou na filmu sehrál, v případě herců i jméno postavy.

Práce s databází probíhá prostřednictvím tříd. Na druhou stranu tyto třídy jsou jediné místo, kde se s databází pracuje. To šetří námahu a aplikaci výrazně zpřehledňuje. Popíši třídu Filmy, struktura a práce s ostatními (tj. s třídami Autori, Uziv a Ciselnik) je analogická.

Třída kompletně zapouzdřuje práci s tabulkou filmy a tabulkami s ní spojenými, tj. ne například seznam filmů nebo jeden konkrétní film. Obsahuje několik proměnných:

- `$filmy` – pole filmů, které je přístupné po vyhledávání nebo po vytvoření seznamu
- `$film` – asociativní pole představující jeden film, kromě samotného záznamu z tabulky filmy může obsahovat i pole s odpovídajícími záznamy z tabulek autori, keywords, zanry a dalších

- `$pocet` – počet všech záznamů v tabulce, které odpovídají nějakému dotazu

- `$krok` – počet záznamů na stránku, tj. maximální počet záznamů, které se načítají do pole `$filmy`

Metody třídy Filmy zmíním jen ty nejpodstatnější:

- `PripravSeznam()` – připraví pro další použití pole `$filmy` v závislosti na parametrech. Parametr `$od` určuje začátek intervalu záznamů, které nás zajímají, `$sort` říká, podle jakého atributu se mají záznamy třídit, `$filtr` je řetězec počátečních znaků názvu filmu, `$zanr` omezuje hledané filmy jen na konkrétní žánr. SQL dotaz se v závislosti na těchto parametrech postupně sestaví (je-li například definován žánr, provede se LEFT JOIN s tabulkou filmy_zanry) a získané záznamy se uloží do pole. Navíc se provede dotaz na celkový počet odpovídajících záznamů v databázi a uloží do proměnné `$pocet`.

- `PripravSeznamRegEx()` – podobná metoda, jako předchozí, jen vrací záznamy v závislosti na shodě atributů soubor a nazev se zadanými regulárními výrazy.

- `PripravFilm()` – získá z databáze a připraví do asociativního pole `$film` jeden konkrétní film podle primárního klíče.

- `PridejFilm()` – přidá do databáze nový film. Jednak samotný záznam do tabulky filmy, jednak záznamy do tabulek filmy_zeme a filmy_jazyky (protože jazyky a země odpovídající filmu se editují jinak než je tomu u dalších číselníků – současně s „obyčejnými“ atributy filmu – je aktulizace zmíněných tabulek součástí této metody). Pokud se některý záznam nepodaří vložit, metoda se pokusí smazat záznamy již vložené a vrací `false`.

- `UpravFilm()` – upraví film v databázi. Ze stejných důvodů, jako předešlá metoda, pracuje jednak se záznamem v tabulce filmy, jednak se záznamy v tabulkách filmy_zeme a filmy_jazyky.

- `UpravSoubor()` – nastaví cestu k videosouboru příslušného filmu

- `Hledej()` – vyhledává filmy podle zadaných kritérií. Tato kritéria vesměs odpovídají atributům tabulky filmy, nemá proto smysl je zde

rozebírat. Vyjímkou je parametr `$autor`, který umožňuje hledat i podle začátku jména autora. Je-li tento parametr neprázdný, provádí se proto LEFT JOIN s tabulkou filmy_autori a další LEFT JOIN s tabulkou autori. Výsledek dotazu se opět ukládá do pole `$filmy`.

- `PripravAutory` – naplní prvek `autori` asociativního pole `$film` záznamy o autorech příslušného filmu. Provádí se dvakrát LEFT JOIN tabulky filmy_autori postupně s tabulkami autori a role. Tato metoda se využívá při práci s autory konkrétního filmu

- `PridejAutora`, `SmazAutora` – přidává, resp. maže autora konkrétního filmu.

Několik dalších metod slouží k obdobným účelům jako poslední tři zmíněné, jen se týkají jiných tabulek. Zajímavé jsou až poslední dvě metody:

- `FilmyPodleKeywords()` – funkce, která se využívá při procházení filmů pomocí klíčových slov. Ukládá do pole `$filmy` všechny filmy, které obsahují všechna, v parametru funkce zadaná, klíčová slova. Tohle je jeden z příkladů, kdy by se hodila nějaký „vyspělejší" databáze, než MySQL 4.0, která neumí průnik SELECTů pomocí INTERSECT a musíme se tedy na každé klíčové slovo zeptat zvlášť a průnik filmů dělat až aplikačně.

- `KeywordsPodleFilmu()` – obdobná funkce, která se také používá při procházení pomocí klíčových slov, tentokrát pro určení množiny klíčových slov, které odpovídají aspoň jednomu filmu. Na to už samozřejmě stačí jednoduchý dotaz, opět se ale počítá průnik množin klíčových slov jednotlivých filmů, aby se mohla vyřadit ta, která odpovídají všem filmům (funkce se pouužívá pro zúžení množiny odpovídajících filmů výběrem dalšího klíčového slova, takže slova odpovídající všem filmům nepotřebujeme).

3.4 Řešení multijazykovosti

Jedním z hlavních cílů bylo navrhnout aplikaci tak, aby šla snadno lokalizovat, zcela bez nutnosti zasahovat do zdrojových kódů.

Řešení, které jsem použil, je velmi jednoduché. Veškeré texty, které jsou „napevno" ve zdrojových kódech, ať už přímo v HTML kódu nebo v PHP

skriptech, jsou definovány jako řetězcové konstanty ve zvláštních souborech v adresáři `lang`. Příkladem budiž následujících několik řádek:

```
define("STR_FAST_SEARCH", "Rychlé hledání");
define("STR_FAST_SEARCH_DESC",
        "hledá v názvech filmů a jménech autorů");
define("STR_FILE_NOT_FOUND", "Soubor nenalezen");
define("STR_AT_MOST", "nejvýše");
define("STR_APPROXIMATELY", "přibližne");
define("STR_MOVIE_NAME_CONTAINS", "Název filmu obsahuje");
define("STR_NEW_SEARCH", "Nové hledání");
define("STR_OK", "OK");
```

Aby se předešlo kolizím, mají všechny tyto „jazykové" konstanty předponu `STR`. Ve zdrojových kódech jsou pak vypisovány jen hodnoty takto definovaných konstant. Jazyků může tímto způsobem definováno libovolné množství a v samotné aplikaci mezi nimi lze přepínat. Tato možnost ale nebude mít praktické uplatnění, neboť samotný obsah, tj. texty uložené v databázi, vlastnost multijazykovosti nemají.

3.5 Databázová abstrakce

Jedním z cílů je, aby aplikace nebyla závislá na konkrétním databázovém serveru. Přesto by však bylo nepřiměřeně složité snažit se o pokrytí všech rozšířených databází, zejména těch komerčních (například Oracle), nebo omezených na konkrétní platformu (MS SQL od Microsoftu). Cíl, který si kladu, je o něco skromnější – možnost použít libovolnou rozšířenou databázi, která je volně šiřitelná (ideálně open source) a multiplatformní. Tomu vyhovují následující tři:

- **MySQL**, nejoblíbenější a nejrozšířenější open source databázový systém, kterému byla ale dlouhou dobu vytýkána nedostatečná implementace jazyka SQL a podpory transakcí, což se pozitivně změnilo s příchodem verze 4.1 a dále verze 5, ty však nejsou dosud dostatečně rozšířené

- databáze **PostgreSQL**, jejíž podpora u hostingových služeb není o mnoho menší, než v případě MySQL, nicméně co do rozšířenosti ve skutečném využití má co dohánět. Přitom co do podpory standardů SQL92 a SQL99 byla dlouhou dobu daleko před MySQL.

- **SQLite**, malá souborová databáze, která však podporuje všechny potřebné konstrukce jazyka SQL.

Databázová abstrakce je implementována prostřednictvím třídy DB. Co se týče MySQL, pro kterou existují v PHP dvě rozšíření – původní, *mysql* pro databáze do verze 4.0 a *mysqli* pro verze 4.1 a vyšší, i na to třída DB pamatuje, ikdyž v této aplikaci toto rozdělení využití nenajde.

Třída DB je ve své podstatě velmi jednoduchá. Obsahuje proměnnou `$type` udávající typ databázového serveru (možné hodnoty jsou mysql40, mysql40, sqlite a pgsql) a dále proměnné, ve kterých se uchovávají údaje pro připojení do databáze (`$host`, `$port`, `$user`, `$password` a `$db`). Všechny tyto proměnné se nastavují v konstruktoru. Nakonec je zde proměnná `$dba`, sloužící jako identifikátor spojení pro databáze PostgreSQL, SQLite a MySQL ve variantě `mysql41`.

Metody odpovídají běžným elementárním operacím s databází:

- `connect()` se připojuje k databázi na základě proměnných nastavených v konstruktoru
- `query()` provede jeden dotaz a vrátí jeho výsledek
- `fetch_array()` vrátí další řádek výsledku dotazu SELECT jako asociativní pole
- `num_rows()` vrací počet záznamů vrácených dotazem SELECT
- `insert_id()` vrací poslední hodnotu primárního klíče s vlastností auto_increment
- `result()` vrací hodnotu konkrétního atributu z prvního řádku vráceného dotazem SELECT

Jejich tělo (kromě poslední jmenované) tvoří vždy struktura `switch`, která se rozhoduje podle proměnné `$type`, tedy podle použitého databázového systému, a na základě toho volá funkce přímo určené pro práci s konkrétním databázovým serverem.

3.6 Řešení distribuce přes BitTorrent

Komplexní automatizované řešení distribuce videa sítí BitTorrent jsem nakonec pro nepřiměřenou složitost zavrhl. Bylo by nutné po vytvoření torrentu „dát nějak vědět" klientu, že má začít videosoubor seedovat. Klient by musel jít ovládat z PHP skriptu, musel by tedy běžet v podstatě jako služba na pozadí a komunikace s ním by musela probíhat nějakým zvláštním protokolem (vzhledem k PHP skriptu by se takový klient vlastně choval jako server). Žádný takový BitTorrentový klient jsem ale neobjevil, proto se budeme muset spokojit s následující funkcionalitou.

Součástí aplikace je jednoduchý BitTorrent Tracker. Jde o část (mírně upravených a zjednodušených) zdrojových kódů komplexnějšího trackeru TorrentTrader v1.0.3 LITE, který byl vydán pod licencí GPL. Předpokládáme, že administrátor aplikace bude mít klient puštěný jako uživatelskou aplikaci. V něm „ručně" vytváří k videosouborům torrenty s adresou našeho trackeru a poté je zveřejňuje v katalogu.

3.7 Řešení podcastingu

Vzhledem k tomu, že podcastový kanál je v podstatě obyčejný RSS kanál, navíc je jen adresa audio či videosouboru, zužuje se problém implementace podcastingu na problém implementace RSS kanálů. RSS kanál je dynamicky generovaný XML dokument s danou strukturou. Zájemce o přesnou specifikaci mohu odkázat na [2]. Pro podcasty je přitom důležitý zejména element `<enclosure />`, který obsahuje URL adresu samotného videosouboru.

Podcastové kanály generuji v souboru `rss_pre.php`. Na něj mohu odkazovat jako na kteroukoliv jinou sekci webu, protože tento soubor vkládá ještě před odesláním hlaviček HTML dokumentu. Pokud odešlu HTTP hlavičku `Content-type: text/xml`, můžu dále tvořit obsah XML dokumentu a využít toho, že už proběhla úvodní část skriptu `index.php`, takže jsem připojen k databázi a mám vložené všechny důležité konstanty, proměnné a funkce.

Je možné definovat v zásadě libovolný počet různých podcastových kanálů. Definují se v konfiguračním souboru `config.ini.php` jako prvky pole `$PODCAST_FEEDS`. Tyto první, které definují jednotlivé kanály jsou asociativní pole s klíči name, num, file_regex a name_regex. Hodnota prvního znamená název kanálu, druhého pak počet podcastů v tomto kanálu. Třetí a čtvrtý prvek jsou regulární výrazy, podle kterých se matchuje cesta k video-

souboru, resp. název videa. Jde tak například snadno vytvořit kanál pro videosoubory různých formátů, nebo kanál pro soubory umístěné v konkrétním adresáři. Dále záleží hlavně na konvenci, jakou jsou soubory pojmenovávány.

Kapitola 4

Závěr

Videosoubory lze na Internetu distribuovat různými způsoby, z toho některé jsou pro multimediální data vhodnější, některé méně. Ukázalo se, že i bez specializovaných služeb, jako jsou různé mediální či streamovací servery, lze video na internetu celkem snadno nabízet, ale musíme si vesměs vystačit s distribucí pomocí protokolu HTTP, který není pro přenos větších objemů dat právě nejvhodnější, navíc pro přenos multimediálních dat prakticky neposkytuje žádné výhody.

Pokud bychom chtěli navrhnout komplexní řešení, už bylo velmi dobré vycházet z konkrétní platformy (to se týká zvláště streamovacích serverů, které jsou vesměs platformě závislé) a aplikaci psát více „na míru". Efektivní použití P2P sítí pro distribuci se také ukazuje jako poměrně složité a platformně závislé.

Katalog je dostatečně univerzální na to, aby jej bylo možné použít ve většině aplikací spíše menšího rozsahu. Jeho univerzálnost se může ale stát nevýhodou, pokud požadujeme nějaké zvláštní vlastnosti, které by bylo obtížnější takto univerzálně implementovat.

Příloha A

Uživatelská dokumentace

V této kapitole popíši práci s aplikací z uživatelského pohledu, tj. především její ovládání. Nebudu přitom předpokládat ze strany uživatele žádné hluboké znalosti.

A.1 Struktura hlavní stránky

Protože aplikaci lze skinovat a různě konfigurovat, může se rozložení stránky lehce, i poměrně výrazně, lišit právě v závislosti na vybraném grafickém skinu, navíc některé popisované vlastnosti nemusí být administrátorem povolené. Budu vycházet z původního skinu a nastavení, kde je hlavní menu na levé straně a žádné části aplikace nejsou skryté nebo zakázané. Navíc budu samozřejmě předpokládat původní české jazykové nastavení.

Kromě záhlaví je stránka rozdělena na hlavní menu, umístěné vlevo, a hlavní obsahovou část, která se samozřejmě mění v závislosti na tom, v jaké sekci webu se právě nacházíme. Jednotlivé sekce budeme nyní procházet postupně tak, jak jsou umístěny v hlavním menu. V první části vynecháme veřejně nepřístupné, administrační, sekce.

A.2 Procházení filmů

Pod odkazem **Procházet filmy** se skrývají další tři podsekce.

Základní varianta je procházení filmů podle jednoho nebo dvou počátečních písmen. První písmeno lze rovnou na této stránce vybrat kliknutím na něj. Kliknutím přímo na odkaz **Procházet podle počátečních písmen**

názvu se žádná podmínka nespecifikuje, tj. zobrazují se všechny filmy. Počáteční, případně druhé, písmeno je možné upřesnit či změnit později. Výsledný seznam je pak možné třídit podle různých sloupců a procházet po stránkách. Kliknutím na název filmu se zobrazí jeho detail.

Druhá možnost, která se skrývá pod odkazem **Procházet podle žánrů** je vlastně rozšíření té první, kdy lze navíc dopředu specifikovat konkrétní žánr, podle kterého se budou filmy filtrovat.

Nejzajímavější je potom možnost **Procházet podle klíčových slov**. Kliknutím přímo na tento odkaz se zobrazí seznam všech klíčových slov, dělený podle počátečního písmena, kliknutím na písmeno se zobrazí se ukáže jen seznam těch klíčových slov, které na něj začínají. Výběrem konkrétního klíčového slova ze seznamu se potom zobrazí stránka, kde na levé straně jsou odpovídající filmy a na pravé klíčová slova, kterými lze zúžit výběr. Jsou to klíčová slova, která jsou přiřazena alespoň k jednomu filmu ze seznamu vlevo (vynechaná jsou ta, která jsou přiřazena ke všem filmům, protože jejich výběrem by se výčet filmů nezúžil).

Vybíráním dalších klíčových slov se tedy zužuje výčet zobrazených filmů. Vypisují se jen ty, které obsahují všechna vybraná klíčová slova. Vybrané klíčové slovo lze následně opět odebrat kliknutím na (-) vpravo od něj.

Seznam filmů vyhledaných pomocí klíčových slov již nejde stránkovat ani třídit. Kliknutím na název se opět dostaneme na detail filmu

A.3 Vyhledávání filmů

Pod odkazem **Hledat filmy** najdeme formulář pro vyhledávání filmů podle následujících kritérií:

- Název filmu obsahuje
- Jméno autora začíná na
- Délka
- Rok
- Popis obsahuje

U kritéra **délka** lze navíc zvolit, jestli se má hledat přesná délka, přibližná nebo filmy s vyšší či nižší délkou, než zadanou. Podobně u kritéria **rok**. Vyhledané filmy lze pak procházet stejným způsobem jako při procházení pomocí počátečních písmen (tj. po stránkách s možností třídění).

A.4 Podcast kanály

V této sekci jsou vypsané podcastové kanály, které administrátor nadefinoval. Každý z nich je odkaz na příslušný dynamicky generovaný kanál, takže lze příslušnou URL adresu vložit přímo do podcastového klienta.

A.5 Procházení a vyhledávání autorů

Autoři se procházejí (odkaz **Procházet autory**) obdobně jako filmy podle počátečních písmen. Stejně jako filmy jdou třídit podle různých sloupců a prohlížet po stránkách. Jméno autora je přitom odkazem na stránku s detailními informacemi o něm, včetně jeho filmografie. Vyhledávání autorů (odkaz **Hledat autory**) je také obdobou hledání filmů. Vyhledávací kritéria jsou následující:

- Jméno obsahuje
- Datum narození
- Národnost
- Životopis obsahuje

Podle data narození lze podobně jako podle délky a roku u filmů, hledat jak přesně, tak přibližně.

A.6 Nastavení

V sekci **Nastavení** lze upravit prostředí aplikace. Konkrétně je možné změnit

- **Skin**, tj. grafickou podobu stránek, je-li jich vytvořených několik
- **Jazyk**, kterým aplikace komunikuje. Toto nastavení nemá vliv na to, v jakém jazyku bude zobrazován samotný obsah (tj. popisy filmů a pod.)

A.7 Rychlé hledání

Rychlé hledání je jednoduchý, ale často dostatečně účinný způsob, jak najít požadovaný film nebo autora. Zadaný výraz se vyhledává v názvech filmů a jménech autorů. Výstupem jsou jen odkazy na detail nalezených záznamů. Hledání je opravdu jednoduché, proto ani neposkytuje stránkování. Pokud je záznamů nalezeno větší množství, je vypsáno jenom prvních několik. V takovém případě je dobré zkusit zadat delší výraz, nebo použít pokročilé vyhledávání (sekce **Hledat filmy** a **Hledat autory**).

A.8 Přihlašování, administrační sekce

Máte-li od administrátora přiděleno uživatelské heslo a jméno, můžete se přihlásit do aplikace a provádět administraci obsahu. Je možné, že nebudete mít přístup do všech administračních sekcí, nebo nebudete mít oprávnění provádět některé akce. Taková oprávnění definuje u každého registrovaného uživatele administrátor. Přihlašuje se pomocí formuláře pod hlavním menu. Po zadání uživatelského jména a hesla a odeslání se zobrazí v hlavním menu nové položky. Místo formuláře je pak pod hlavní nabídkou napsáno, kdo je přihlášen, a je zde i odkaz pro případné odhlášení (odhlášení jinak proběhne po určité době – tu definuje administrátor – z bezpečnostních důvodů automaticky).

A.9 Přidávání a upravování filmů

Nový film do katalogu přidáme pomocí formuláře, který se skrývá pod odkazem **Přidat film** v části **Filmy** hlavního menu. Pole **název filmu**, **délka**, **rok** a **popis** není třeba rozebírat. Následuje možnost výběrů jazyků a zemí. Jednu položku vybereme prostým kliknutím, máme-li jich vybrat víc, musíme si pomoct klávesnicí. Současným stiskem klávesy `Ctrl` a kliknutím položku přidáme nebo odebereme z výběru. Pokud podržíme `Shift` a klikneme na položku, označíme celý interval interval od položky naposledy vybrané. Po odeslání formuláře se zobrazí stejná stránka, navíc má však zcela nahoře další nabídku. Film už je uložen v katalogu, máme však ještě možnost nastavit další jeho vlastnosti. Po kliknutí na jednotlivé odkazy se ukáže nové okno, kde je možné tyto vlastnosti určit.

Ještě než se na tyto odkazy podíváme podrobně, řekneme, jak se záznamy

o filmech upravují. Je-li uživatel přihlášen, zobrazuje se při procházení filmů u každého z nich a také na stránkách s detaily jednotlivých filmů, tlačítko **upravit**. To vede na stejný formulář, kterým se filmy přidávají. Hodnoty jsou vyplněny, je možné je změnit a potvrdit odesláním formuláře. V horní části stránky jsou stejné odkazy jako po přidání filmu.

Přiřazení autorů k filmu Klepnutím na odkaz **Autoři** se objeví okno, kde je možné přiřazovat autory k vybranému filmu. Jsou-li už k filmu nějací autoři přidáni, zobrazí se jejich seznam, každého je možné odstranit tlačítkem **smazat**. Nového uživatele k filmu přiřadíme formulářem v dolní části okna.

Přidání obrázků, screenshotů Okno pro práci se screenshoty filmu, které se zobrazí po kliknutí na odkaz **screenshoty**, obsahuje již dříve uložené obrázky a formulář pro přidávání nových. U každého obrázku je možné změnit popisek a potvrdit tlačítkem **OK** nebo jej zcela odstranit kliknutím na tlačítko **smazat**. Nové obrázky se přidávají pomocí formuláře na konci stránky. Je třeba vybrat soubor so obrázkem na lokálním počítači, případně napsat nějaký popisek a odeslat. Je-li vybraný obrázek moc velký, je jeho velikost automaticky zmenšena a je i vytvořena jeho menší verze.

Výběr videosouboru, který už je uložen na serveru Máme-li už videosoubor, který chceme k filmu přiřadit, už uložený na serveru ve správném adresáři (adresář definuje administrátor), můžeme ho v okně, které se zobrazí po kliknutí na odkaz **Vybrat video na serveru**, vybrat. V prvním řádku je napsaná cesta k aktuálně vybranému videosouboru (je-li takový), ve druhém potom aktuální adresář. Následuje výpis aktuálního adresáře. Adresáře jdou procházet, tj. klikneme-li na adresář (jsou uzavřené do hranatých závorek), aktuální adresář se změní. Kliknutím na soubor se tento soubor vybere.

Upload videosouboru Druhou možností, jak video k filmu přiřadit, je jeho upload z lokálního počítače. Okno pro upload videa se otevře kliknutím na odkaz **HTTP upload videosouboru**. Nejdříve průchodem adresáři zvolíme ten adresář, do kterého chceme video uložit. Poté ho ve formuláři v dolní části vybereme na lokálním počítači a odešleme. Maximální velkost souboru, který jde uploadovat, bývá na serveru omezena (příslušná hodnota

je napsána pod formulářem), proto je vhodnější soubor nahrát na server vhodnějším způsobem (třeba přes FTP klienta) a vybrat prvním způsobem.

Upload torrentu Chceme-li video nabízet ke stažení prostřednictvím sítě BitTorrent, uploadujeme příslušný torrent v okně, které se otevře po kliknutí na odkaz **Upload torentu**. V okně je napsáno, jestli už byl nějaký torrent k filmu přiřazen a máme také možnost torrent odstranit.

Nastavení klíčových slov Okno pro práci s klíčovými slovy zobrazíme kliknutím na odkaz **Klíčová slova**. V levé části je výčet slov, která už byla k filmu přiřazena (každé z nich je možné odebrat stiskem tlačítka **smazat**). Přidávat nová slova je možné dvěma způsoby. První z nich je výběr slova v pravé části podle prvního písmena. Kliknutím na písmeno se vypíší všechna klíčová slova, která na něj začínají. Pokud vybereme konkrétní slovo a stiskneme tlačítko **přidat**, zařadí se do seznamu vlevo. Druhá možnost je přidání slova pomocí formuláře dole. Tímto způsobem můžeme přidat i nové klíčové slovo (automaticky se přidá do databáze), musíme mít ale příslušné oprávnění.

Zařazení filmu do žánrů Konečně pod odkazem **Žánry** se skrývá okno pro zařazení filmu do žánrů. Vše je velmi podobné přidávání klíčových slov, jen jednodušší. V pravo je seznam všech existujcích žánrů, vlevo potom žánry přiřazené k filmu.

A.10 Číselníky

Číselníky jsou jednoduché seznamy bez dalších atributů, už jsme se se všemi setkali v předešlém povídání. Jde o **žánry**, **klíčová slova**, **jazyky**, **země** a **filmové role**. V stejnojmené části hlavní nabídky jsou odkazy na administraci každého z nich. Práce s nimi je shodná. Na začátku stránky je formulář pro přidání nového prvku, následuje formulář, ve kterém můžeme zadat několik počátečních písmen tzv. *filtru*. Je-li fitr zadán, zobrazují se v seznamu položek, který následuje, jen výrazy, které na tuto skupinu písmen začínají. Je-li položek mnoho, jsou rozděleny do stránek, mezi kterými se lze pohybovat.

adresář, do kterého se ukládají videosoubory.

Dále je třeba vytvořit databázi a jednotlivé tabulky. V ZIP archivu je SQL skript `db.sql`, který strukturu vytvoří. K dispozici i skript `db.php.sql`, který provede totéž a lze použít, pokud nemáme přístup ke konzole databázového serveru. Je ale třeba nejdříve nastavit proměnné pro připojení do databáze v souboru `config/db.ini.php`.

B.3 Konfigurace

Nastavení aplikace se provádí v souboru `conf/config.ini.php`. Vesměs jde o konstanty, které je třeba nadefinovat, v případě rozsáhlejších struktur je nastavení reprezentováno asociativním polem (konstanty mohou mít jen skalární hodnoty). Podíváme se postupně na jednotlivé možnosti nastavení:

`CFG_LOGIN_TIMEOUT`	*celé číslo*	doba, za kterou dojde k automatickému odhlášení
`CFG_ROOT`	*řetězec*	cesta ke kořenovému adresáři aplikace
`CFG_URL`	*řetězec*	URL adresa aplikace
`CFG_VIDEO_ROOT`	*řetězec*	cesta k adresáři s videosoubory
`CFG_HOMEPAGE`	*řetězec*	sekce, která bude úvodní stránkou
`CFG_COOL_URI`	*true/false*	povoluje/zakazuje použití *hezkých URL*
`$LANGUAGES`	*pole*	seznam jazyků, které se nabízejí na výběr uživateli
`$STYLES`	*pole*	seznam skinů, které se nabízejí na výběr uživateli
`CFG_DATE_FORMAT`	*řetězec*	formátovací řetězec datumů. Znak `d`, `m`, `Y` jsou po řadě nahrazeny dnem, měsícem a rokem.
`CFG_AUTHORS`	*true/false*	kompletně povoluje/zakazuje v katalogu práci s autory
`CFG_AUTHORS_LIST`	*true/false*	povoluje/zakazuje procházení autorů
`CFG_AUTHORS_SEARCH`	*true/false*	povoluje/zakazuje vyhledávání autorů

CFG_MOVIES_LIST	*true/false*	povoluje/zakazuje procházení filmů
CFG_MOVIES_SEARCH	*true/false*	povoluje/zakazuje vyhledávání filmů
CFG_SCREENSHOTS	*true/false*	povoluje/zakazuje přidávání screenshotů k filmům
CFG_SCREENSHOT_WIDTH	*celé číslo*	šířka, na kterou se mají screenshoty zmenšovat v pixelech
CFG_SCREENSHOT_HEIGHT	*celé číslo*	výška, na kterou se mají screenshoty zmenšovat v pixelech
CFG_THUMB_WIDTH	*celé číslo*	šířka zmenšené verze screenshotu v pixelech
CFG_THUMB_HEIGHT	*celé číslo*	výška zmenšené verze screenshotu v pixelech
CFG_KEYWORDS	*true/false*	povoluje/zakazuje používání klíčových slov
CFG_GENRES	*true/false*	povoluje/zakazuje používání žánrů
CFG_LANGUAGES	*true/false*	povoluje/zakazuje přiřazování jazyků k filmům
CFG_COUNTRIES	*true/false*	povoluje/zakazuje přiřazování zemí k filmům
CFG_DOWNLOAD	*true/false*	povoluje/zakazuje přímý download videosouborů
CFG_PODCASTING	*true/false*	povoluje/zakazuje podcasting videosouborů
$PODCAST_FEEDS	*pole*	definuje jednotlivé podcastové kanály; podrobnosti přímo v konfiguračním souboru
CFG_BITTORRENT	*true/false*	povoluje/zakazuje použití sítě BitTorrent pro šíření souborů
CFG_STREAMING	*true/false*	povoluje/zakazuje streaming souborů
CFG_PLAYER_WIDTH	*true/false*	šířka přehrávače streamovaného souboru
CFG_PLAYER_HEIGHT	*true/false*	výška přehrávače streamovaného souboru

Literatura

[1] Bittorent Protocol Specification v1.0.
http://wiki.theory.org/BitTorrentSpecification

[2] RSS 2.0 Specification. http://www.rssboard.org/rss-specification