UNIVERZITA PARDUBICE
Fakulta elektrotechniky a informatiky

Programování video serveru

Zdeněk Špulák

Bakalářská práce
2010

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: **2009/2010**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Zdeněk ŠPULÁK**
Osobní číslo:	**I07971**
Studijní program:	**B2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Programování video serveru**
Zadávající katedra:	**Katedra informačních technologií**

Zásady pro vypracování:

Cílem práce je srovnání současných možností publikace videa na internetu a vytvoření webového videoserveru, kde budou inteligentním způsobem řazena videa. Ta budou také k dispozici v ruzných kvalitách. Vlastní videoserver bude porovnán s dostupnými.
Teoretická část
- vysvětlení pojmu videoserver
- historie videoserveru
- analýza konvertování - použité kodeky, rychlost, ...
- náročnost a zatížení hardware - traffic, vytížení CPU v závislosti na návštěvnosti

Praktická část
- konvertování videa na serveru
- řazení videí do kategorií
- přehrávání videa na stránce - bez nutnosti stahování
- řazení videí podle data, počtu zobrazení a hodnocení
- výběr kvality přehrávaného videa

Rozsah grafických prací:

Rozsah pracovní zprávy:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

- **CASTRO, Elizabeth. HTML, XHTML a CSS : Názorný průvodce tvorbou WWW stránek. 6. vyd. Brno : Computer Press, 2007. 440 s. ISBN 978-80-251-1531-2.**
- **PONKRÁC, Miloslav. PHP a MySQL bez předchozích znalostí : Průvodce pro samouky. 1. vyd. Brno : Computer Press, 2007. 224 s. ISBN 978-80-251-1758-3.**

Vedoucí bakalářské práce: **RNDr. Josef Rak**
Katedra informačních technologií

Datum zadání bakalářské práce: **15. ledna 2010**

Termín odevzdání bakalářské práce: **14. května 2010**

L.S.

prof. Ing. Simeon Karamazov, Dr.
děkan

Ing. Lukáš Čegan, Ph.D.
vedoucí katedry

V Pardubicích dne 31. března 2010

Prohlášení autora

Prohlašuji, že jsem tuto práci vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 12. 5. 2010 Zdeněk Špulák

Anotace

Práce se zabývá možnostmi publikace videa na internetu, vývojem a historií videoserveru. Jsou řešeny problémy, které se videoserverů dotýkají a také jsou popsány požadavky na hardware a software. Cílem praktické práce je vytvoření funkčního videoserveru.

Klíčová slova

video, videoserver, kodek, ffmpeg, FLV, MP4

Title

Programming Video Server.

Annotation

In this paper I was concerned with the problems connected with videos on the internet. I focused especially on the history and development of videoservers and possibilities of publishing videos on the internet. In the Theoretical Part, I also described hardware and software requirements for videoservers. As Practical Part, I created a functional videoserver.

Keywords

video, videoserver, codec, ffmpeg, FLV, MP4

Obsah

Seznam zkratek **8**
Seznam obrázků **9**
Seznam tabulek **9**
1 Úvodní informace **10**
2 Co je to videoserver **11**
2.1 Internetový videoserver 11
3 Historie **12**
3.1 Novodobá éra videoserverů 12
3.1.1 youtube 12
3.1.2 „Kopie“ youtube 13
3.1.3 České videoservery 13
3.2 Technologický vývoj 13
3.2.1 Adobe Flash 13
3.2.2 Microsoft Silverlight 13
3.2.3 HTML 5 13
4 Jak videoserver funguje **15**
4.1 Fungování z pohledu návštěvníka 15
4.2 Principy fungování uvnitř videoserveru 15
4.2.1 Vysvětlení pojmů 15
4.2.2 Základní model fungování 16
4.2.3 Rozšíření základního modelu 17
5 Hardwarové a softwarové požadavky **18**
5.1 Hardware 18
5.1.1 Procesor 18
5.1.2 Pevný disk 18
5.1.3 Operační paměť 18
5.1.4 Shrnutí 18
5.2 Software 19
5.2.1 Operační systém 19
5.2.2 Základní webový server 19

5.2.3 Specifický software 19
5.2.4 Shrnutí 19
6 Zatížení videoserveru v závislosti na počtu uživatelů, následky a jejich řešení 21
6.1 Hlavní problémy 21
6.1.1 Přenos dat (traffic) 21
6.1.2 Vytížení procesoru 21
6.1.3 Nedostatek operační paměti 21
6.2 Příklady z praxe 21
6.2.1 videotube.sk 21
6.2.2 zvraceny.cz 22
6.3 Řešení problémů 25
6.3.1 Přenos dat (traffic) 25
6.3.2 Vytížení procesoru 26
6.3.3 Nedostatek operační paměti 27
7 Vlastní řešení videoserveru 28
7.1 Volba formátu videa 28
7.1.1 FLV 28
7.1.2 VP6 29
7.1.3 MP4 29
7.1.4 H.264 30
7.2 Volba audio formátu 30
7.2.1 MP3 30
7.2.2 AAC 30
7.3 Srovnání kvality videokodeků 31
7.3.1 Formát FLV s kodekem VP6 32
7.3.2 H.264 v kontejneru MP4 32
7.3.3 Časy konvertování 34
7.4 Popis webové stránky videoserveru 35
7.4.1 Layout, rozvržení a grafika 35
7.4.2 JW Player 37
7.4.3 Databáze 39
7.5 Proces konvertování videa 40
7.5.1 Zpracování nahraného souboru 40

7.5.2 Konvertovací skript....41
7.5.3 Konvertování programem FFmpeg....42
7.5.4 Úprava FLV videí....44
7.5.5 Generování náhledů videa....45
7.6 Streamování videa....45
7.6.1 Shrnutí....45
7.7 Návrhy na vylepšení....45
7.7.1 Ochrana videosouborů....46
7.7.2 Generování více náhledů....46
7.7.3 Zpoplatnění obsahu....46
8 Porovnání s jinými videoservery....48
8.1 Uživatelské srovnání....48
8.2 Technické srovnání....48
8.3 Shrnutí....49
9 Závěr....50
Literatura....51
Příloha A – Elementární kód pro vložení JW Playeru na stránku....53
Příloha B – SQL dotaz používaný pro vyhledávání....54
Příloha C – Adresářová struktura....55
Příloha D – kód pro generování náhledů z videa....56

Seznam zkratek

AAC	Advanced Audio Coding
AJAX	Asynchronous JavaScript and XML
AMD	Advanced Micro Devices
API	Application Programming Interface
ASP	Active Server Pages
AVI	Audio Video Interleave
CSS	Cascading Style Sheets
DRM	Digital rights management
DVD	Digital Video Disc
FLV	Flash Video
Fps	Frames per second
HD	High Definition
HTML	HyperText Markup Language
ISO	International Organization for Standardization
MIME	Multipurpose Internet Mail Extensions
MP3	MPEG-1 Audio Layer 3
MP4	MPEG-4 Part 14
OS	Operační systém
PHP	Hypertext Preprocessor
SEO	Search engine optimization
SQL	Structured Query Language
VOB	Video Object
XML	Extensible Markup Language

Seznam obrázků

Obrázek 1 – Statistika návštěvnosti videoserveru videotube.sk (březen 2010) 22
Obrázek 2 – Statistika návštěvnosti videoserveru zvraceny.cz (duben 2010) 23
Obrázek 3 – Přenos dat na datovém serveru zvraceny.cz 24
Obrázek 4 – Vytížení procesoru na datovém serveru zvraceny.cz 24
Obrázek 5 – Využití operační paměti na datovém serveru zvraceny.cz 25
Obrázek 6 - Originální video 31
Obrázek 7 – FLV s kodekem VP6 a datový tok 2000 kbit/s 32
Obrázek 8 – MP4, kodek H.264, datový tok 500 kbit/s, profil max 33
Obrázek 9 – Kodek H.264 v MP4 kotejneru, datový tok 2000 kbit/s, profil ultrafast 34
Obrázek 10 – JW Flash Player 5.1 38

Seznam tabulek

Tabulka 1 – Časy konvertování 35

1 Úvodní informace

Cílem teoretické části je obeznámení s pojmem videoserveru jako takovým, dále pak jeho vývoj a historie. Je pojednáno o principech a fungování, popsány problémy, které se videoserverů přímo dotýkají a naznačeno, jak tyto problémy úspěšně řešit. Zároveň se tato práce zabývá softwarovými a hardwarovými požadavky a také zatížením v závislosti na počtu uživatelů.

V praktické části je cílem vytvořit funkční videoserver, kde budou videa přehrávána přímo na stránce bez nutnosti jejich stažení. Zároveň tato videa logicky dělit do kategorií, řadit je na stránce podle různých kritérií a umožnit uživatelům nová videa nahrávat.

2 Co je to videoserver

Pod pojmem videoserver si lze představit mnoho zařízení, služeb i technologií. Asi jako první nás napadne internetová stránka obsahující nějaká videa. Práce, kterou čtete, se bude zabývat právě tímto pojmem. Avšak není to pouze internetová stránka, co se za tímto názvem skrývá. Videoserver může být zařízení uchovávající bezpečnostní nahrávky, může to být úložný prostor pro televizní stanice, které si sem zálohují filmy a pořady. A nebo také přímo software sloužící pro vysílání videonahrávek.

2.1 Internetový videoserver

Jak již bylo zmíněno, budeme se věnovat videoserveru v podání internetové stránky. Základní princip fungování je relativně prostý. Uživatelům je nabízen video-obsah, který si mohou prohlédnout bez nutnosti stáhnout video do svého počítače. Video je přímo umístěné na stránce, kde jej lze přehrát. Výše popsané je zcela základní model fungování. Pokud bychom si stanovili určité dělení do kategorií, byl by zde videoserver, jehož obsah tvoří

a) majitel/administrátor webu

b) uživatelé

První případ, kdy videa vkládá správce, je hojně využíván u televizních stanic a jejich video-archivů. Fanoušci daného pořadu si pak mohou zpětně na webu prohlédnout díl, který nestihli v televizi apod. Velký boom nastává na zpravodajských serverech, kde se vedle textu se zprávou objevují čím dál častěji krátká videa s výpovědí svědků, tiskových mluvčí atd. Druhou možností je pak umožnit nahrávat videa přímo uživatelům webu. Obecně je tento model stránek, kdy web tvoří sami uživatelé, označován jako web 2.0. Rozhodně nejznámější je portál youtube, který měsíčně navštíví přes neuvěřitelných 70 milionů lidí [1].

Videoservery – v čele s youtube – jsou v dnešní době velice populární. S rozmachem mobilních telefonů obsahujících možnost nahrávat video, jsou více a více vyhledávanou službou na internetu. Není totiž nic snazšího, než pomocí mobilu nahrát video, umístit jej na web a odkaz poslat kamarádům a podělit se s nimi o svůj zážitek.

3 Historie

Historie videoserverů nesahá nijak hluboko do minulosti. Jedná se o relativně mladý způsob publikace na webu. Na počátku éry internetu byly rychlosti připojení velice nízké, a proto byl přenos větších dat (vč. videí) prakticky nemyslitelný. S postupným zvyšováním kvality a navyšováním konektivity internetových přípojek se začal stávat přenos objemnějších dat realitou.

Původním jediným řešením publikace videa bylo nahrát soubor na web a umístit odkaz ke stažení. Toto video si pak uživatel stáhnul a přehrál na svém počítači. Tato metoda je však značně neefektivní, a to hned z několika důvodů. Asi největším problémem je nutnost stažení celého souboru. Přitom uživatele zajímá třeba jenom část videa, nikoliv celek – vždy však musí stáhnout video celé. Často také nastává situace, kdy je video kódováno pomocí kodeku, který není nainstalován na cílovém počítači.

3.1 Novodobá éra videoserverů

Veliká revoluce přišla roku 2003, když firma Macromedia (dnes Adobe) vydala svoji šestou verzi Flash Playeru [2]. Jedná se o doplněk pro internetový prohlížeč (Internet Explorer, Mozilla Firefox, Gogole Chrome, Safari atd.) umožňující přehrát animace ve formátu swf. V dnešní době je tato platforma hojně využívána především pro hry, přehrávače videí a v neposlední řadě také pro reklamní bannery. V současné době (březen 2010) je dle dostupných informací Flash Player nainstalován na 99 % počítačů připojených k internetu [3]. Zmiňovaný Flash Player 6 umožňoval přehrávání videí ve formátu FLV přímo na webové stránce. Od této doby se videa objevovala na různých stránkách, především propagující produkty firem apod.

3.1.1 youtube

Opravdový boom nastal v květnu 2005, když bývalí zaměstnanci společnosti PayPal , jmenovitě Chad Hurley, Steve Chen a Našed Karin spustili betaverzi portálu youtube. Do oficiálního provozu byl server nasazen v prosinci téhož roku. Obsah se pak rozrůstal doslova raketovým tempem. V červnu 2006 bylo denně nahráno až 65 tisíc nových videí a počet přehrání se blížil ke 100 miliónům. Zhruba po roce po oficiálním spuštění byl youtube zakoupen společností Google za 1,65 miliardy dolarů. Server dále vzkvétal, nová videa stále přibývala bleskovou rychlostí, ale vyskytl se zásadní problém – porušování autorských práv. Mnoho uživatelů nahrává klipy svých oblíbených skupin, televizních pořadů a nebo „jenom“ přidá k videu z dovolené písničku, na která nevlastní autorská práva. Varování na stránkách nepomáhá, proto youtube čelí žalobám a nakonec uzavírá smlouvy s nahrávacími studii. Od února 2009 je automaticky každé video zanalyzováno a pokud obsahuje autorsky neoprávněný materiál, je video smazáno nebo je vypnut zvuk. V současné době je youtube dostupný ve 22 jazykových mutacích, vč. češtiny. V některých zemích je dokonce blokován z důvodu přílišné svobody slova [4]. Velikou výhodou je volba mezi různými kvalitami přehrávaného videa. Dostupných formátů je mnoho. Od klasického FLV, přes 3GP pro mobilní telefony až po Full HD videa

v plném rozlišení 1920x1080 bodů. Server se stále vyvíjí, nabízí mnoho doplňkových funkcí – komentování videí, možnost zobrazit přehrávač na vlastních stránkách, posílání zpráv mezi uživateli a mnoho dalších vymožeností.

3.1.2 „Kopie“ youtube

Z úspěchů youtube se snažili profitovat i ostatní, a proto vznikalo – a stále vzniká – spousty nových, více či méně podobných, serverů jako je youtube. Vznikají různé odnože a varianty videoserverů. Velmi populární se stává např. tzv. prémiový obsah. Uživatel si zaplatí předplatné (kredit) a jsou mu dostupná videa, která neplatící návštěvníci nevidí. Objevují se weby zaměřené pouze na nějakou kategorii – filmové upoutávky nebo třeba erotická videa. Ti všichni těží z úspěchu youtube.

3.1.3 České videoservery

Konkrétně v České republice je asi nejznámějším videoserverem stream.cz. Předností – a zřejmě i hlavním zdrojem popularity – se staly vlastní pořady. Mezi nejznámější patří „Nekecej“ a „Kalendář Míši a Ríši“. Od července 2008 spolupracuje stream.cz s TV Prima a je možné sledovat produkci této televize online [6].

3.2 Technologický vývoj

3.2.1 Adobe Flash

Po technologické stránce se neudálo nic převratného. Formát FLV patří stále mezi nejpopulárnější. Avšak velice výraznou změnou bylo vydání nového Flash Playeru v srpnu 2007. Konkrétně se jednalo o verzi 9 update 3. Tato verze umožňovala nově podporu pro přehrávání mp4 videí, vč. vysoce kvalitního H.264 kodeku [2]. Srovnání kodeků je obsahem jiné kapitoly.

3.2.2 Microsoft Silverlight

Ani gigant Microsoft nechtěl být pozadu, a proto přisel s vlastní variantou přehrávače. Ten nazval Microsoft Silverlight a první verze byla vydána v dubnu 2007. Obrovskou výhodou tohoto produktu je podpora programovacích jazyků .NET [6]. Pro vývojáře aplikací/animací tedy odpadá nutnost učit se novému jazyku. Nevýhodou je však stále malé povědomí o Silverlightu. Flash je zkrátka v současné době stále dominantní. Zřejmě největším úspěchem Silverlightu v našich končinách bylo zvolení této technologie namísto Adobe Flash při online vysílání XXI. olympijských her z Vancouveru na webu zoh.cz.

3.2.3 HTML 5

Velice zajímavým počinem je pak nová verze HTML jazyka. Nový standard HTML5 totiž obsahuje nový tag <video>. Tento standard podporuje H.264 kodek a zcela svobodný kodek Theora. Bohužel je však tento standard stále v počátečních fázích a po internetu jsou to spíše experimenty než nějaké finální nasazení. Dobrou zprávou však je, že i youtube s tímto standardem experimentuje a umožňuje přepnout web do HTML5. Nespornou výhodou tohoto řešení je absence jakéhokoliv modulu. O všechno se stará

prohlížeč, nic není potřeba doinstalovávat. Z toho také plyne mnohem menší zátěž na systém [7].

4 Jak videoserver funguje

4.1 Fungování z pohledu návštěvníka

Pro uživatele je fungování videoserveru relativně snadné. Zadá do svého internetového prohlížeče adresu, kde se zobrazí nějaký seznam videí. Pokud na požadované video klikne, ukáže se stránka se zabudovaným přehrávačem na stránce a bude možné video zhlédnout. V případě nahrávání nového souboru je situace obdobná. Člověk vstoupí na stránku připravenou k nahrání, vybere soubor ze svého počítače, vyplní nezbytné údaje – název, popis, klíčová slova atd. V poslední fázi pak pouze spustí samotné nahrávání na web. Až se úspěšně nahraje video na server, je uživateli zpřístupněn odkaz, pod kterým je možné jeho dílo zhlédnout a on jej může rozeslat svým kamarádům a známým. To je po uživatelské stránce víceméně princip fungování základního modelu (přehrávání a nahrávání) videoserveru.

4.2 Principy fungování uvnitř videoserveru

Abychom mohli správně pochopit, co se děje uvnitř, tedy to, co uživatel nevidí, je nutné si nejprve vysvětlit pár pojmů.

4.2.1 Vysvětlení pojmů

Kodek je zařízení nebo program umožňující transformaci datového toku nebo signálu. Používají se pro zakódování a následně dekódování dat za účelem přenosu, uchování nebo šifrování dat. Právě slovo kodek je složenina slov kodér a dekodér. Pojem kodek je velmi často zaměňován s formátem videa nebo s kontejnerem (viz níže). Je nutné si uvědomit, že se jedná čistě o posloupnost instrukcí nebo kódu programu, jak data reorganizovat do jiné podoby. Jistě nejlepší vysvětlení bude na příkladu. Jistě všichni známe formát zvuku MPEG Layer 3 (mp3). Kodeky, které umožňují tento formát (de)kódovat, jsou např. LAME, FhG (Fraunhofer-Gesellschaft), Blade nebo Xing. Dalším příkladem mohou být dnes velice populární kodeky DivX a XviD. Oba shodně pracují s videem ve formátu MPEG-4 ASP. To, že oba dva kodedy pracují se stejným formátem mimo jiné znamená, že jsou kompatibilní (video zakódované jedním z nich lze dekódovat pomocí jiného). Další mýlkou je označení dekodéru v přehrávačích „kodekem“. Žádný ze stolních přehrávačů, kteří máme mnozí v obýváku, má dekodér, nýbrž nemá kodér – nepořebuje ho totiž, video je pouze přehráváno (dekódováno) [8]. Kodeky se dělí do dvou základních skupin – ztrátové a bezztrátové kodeky. V případě ztrátových kodeků jsou odstraněna určitá data, čímž dojde ke zmenšení velikosti souboru. Typickým příkladem ztrátových kodeků je audio formát MP3, který ze zvuku odstraňuje pro člověka neslyšitelné frekvence. Výhodou ztrátových kodeků je veliká úspora potřebného místa pro uložení, naopak nevýhodou je ztráta kvality. Opakem jsou bezztrátové kodeky. Zde dochází ke kompresi dat, podobně jako to známe u klasického zip, rar atd. formátu. Video (audio) je pak zkomprimováno, čímž sníží svoji velikost. Obrovskou výhodou je zachování plné kvality originálu. Bezztrátové kodeky se proto hodí pro uchovávání dat např. ve

studiích apod. Nevýhodou je nižší poměr komprese – ztrátové kodeky jsou mnohem účinnější [9].

Multimediální kontejner je velmi často zaměňován právě s kodekem. Kontejner je však „pouze“ jakýsi předpis a určení, jak jsou data uložena a organizována. Díky tomu je možné do jednoho souboru vměstnat video, několik jazykových mutací zvuku a zároveň také titulky ve více jazycích. Kontejner zajistí synchronizaci všech prvků – videa, audia a titulků. Pro přehrání jednotlivých kontejnerů se používá tzv. demuxer (nebo též splitter), který rozdělí datové proudy do různých kodeků a následně do výstupních zařízení. Kontejner sám neříká nic o vnitřní kompresi uložených dat, ta je určena použitým kodekem. Ne každý kontejner je pak kompatibilní s každým kodekem. Dnes asi nejznámější a nejrozšířenějším kontejnerem je AVI. Vyvinula jej společnost Microsoft, v původní verzi bylo video nekomprimované s rozlišením 160x120 bodů a 15 snímků za sekundu. Soubor byl omezen do velikosti max. 1 GB. Následovalo modernizování AVI kontejneru až do současné podoby, kdy je možno použít libovolný kodek a velikost souboru není omezena. Velikou nevýhodou je umístění tabulky s pozicí snímku v souboru až na konci samotného souboru. To znemožňuje využít tento kontejner na internetu – video je nutné celé stáhnout. Mezi další hojně využívané kontejnery patří QuickTime, ASF, MP4 nebo RealMedia. V mobilních telefonech se stal standardem 3GP, u klasických DVD Video disků se využívá VOB. Jedním z nejnovějších a zároveň otevřených formátů kontejneru je Matroska, dnes známější jako přípona souborů mkv. Pro popis svojí struktury využívá jazyk XML, umožňuje nést prakticky jakákoliv data (nejen video a audio), je uzpůsoben i pro titulky, menu a další [10].

4.2.2 Základní model fungování

Samotná funkcionalita videoserveru závisí na návrhu samotného projektu. Jak bylo již několikrát zmíněno, existuje mnoho variant, pomocí kterých lze web realizovat. Avšak postupy jsou vždy stejné. Základní model videoserveru má dvě hlavní funkce – přehrávání a nahrávání videí.

Pro přehrávání je potřeba zajistit přehrávač, který se stane součástí webových stránek. Dnes nejrozšířenější technologií je Adobe Flash, využít lze také Microsoft Silverlight nebo Windows Media Player. Nevýhodou je nutnost, aby uživatel měl nainstalovanou podporu v prohlížeči pro zvolenou technologii. V případě Flashe se uvádí 99% dostupnost všech počítačů připojených k internetu [3]. Takový přehrávač načítá videosoubor ze serveru a přehrává jej uživateli přímo na stránce.

Nahrávání videí je o poznání složitější proces. Uživatel vybere soubor ze svého počítače, vyplní potřebná data – název videa, klíčová slova atd. – a spustí nahrávání. Po dokončení procesu nahrávání je původní video uloženo na serveru a v tuto chvíli se spouští nejdůležitější proces celého videoserveru – konvertování videa. Jedná se o proces, kdy se video, které uživatel nahrál, převádí do formátu, který dokáže přehrát náš přehrávač na stránce. Samotné konvertování je velice náročné na výkon procesoru a serveru zabere dost

času. Po úspěšném převedení do námi požadovaného formátu je video uloženo na server a zpřístupněno ostatním uživatelům pro zhlédnutí.

4.2.3 Rozšíření základního modelu

Současné videoservery se samozřejmě se základním modelem neuspokojí, a proto nabízejí mnoho dalších funkcí a vylepšení. Velice populární je tzv. progress bar, kdy je zobrazen aktuální stav při nahrávání videa a uživatel může v reálném čase sledovat, kolik % dat je již nahráno a odhad času, jak dlouho bude ještě nahrávání trvat. Přínosným vylepšením může být odesíláni e-mailu po dokončení konvertování videa, počítání počtu zobrazení, možnost nechat ostatní uživatele vyjádřit se – tedy komentáře, hodnocení atd. Fantazii se meze nekladou a lze vymyslet prakticky cokoliv, stěžejním procesem však vždy bude konvertování videa do požadovaného formátu.

5 Hardwarové a softwarové požadavky

Provozování videoserveru není nikterak jednoduchá záležitost. Už v prvních krocích je nutné opatřit si výkonný server se specifickými požadavky. Běžně dostupný webhosting bohužel ve většině případů nestačí.

5.1 Hardware

5.1.1 Procesor

Po stránce hardwaru patří mezi hlavní požadavky výkon procesoru. Obecně platí, že čím výkonnější, tím lepší. Konvertování videa je velice náročné právě na procesor. Zde se určitě vyplatí si připlatit, výkon procesoru je pak k nezaplacení a při konvertování to bude znát.

5.1.2 Pevný disk

Videa se musí někam ukládat, proto je potřeba poměrně hodně ukládacího prostoru. Videa patří k nejvíce objemným souborů na internetu a je třeba mít tuto skutečnost na paměti.

5.1.3 Operační paměť

Výše dva uvedené požadavky jsou tím nejpotřebnějším a nejhlavnějším. Avšak nesmíme zapomenout na velikost operační paměti. S tou bývá problém při přehrávání videí, konkrétně v případě, když jsou na webu dlouhá videa a tedy veliké soubory na disku. U menších videoserverů však tento problém obvykle nenastává.

5.1.4 Shrnutí

Pokud bychom měli být konkrétní a měli si určit hardware, který je potřebný pro provoz videoserveru, situace by byla asi následující.

Jako úplný základ, který poslouží pro naprosto základní videoserver s řádově desítky přehranými a několika málo nahranými videi denně, by byl to server s procesorem o taktu 600 MHz, 256 MB RAM a s 20GB pevným diskem. Jak je vidět, požadavky jsou na dnešní dobu poměrně mírné a splní je i virtuální server z nabídky mnoha hostingových firem.

Naopak konfigurace s moderním čtyřjádrovým procesorem Intel Xeon X3220, 4 GB operační paměti a 1 TB úložného prostoru je zárukou kvalitního serveru, který poskytne dostatečný výkon i pro středně veliké videoservery. Myšleno desítky až stovky nových a tisíce přehraných videí denně.

Samostatnou kapitolou jsou pak větší a velké videoservery, kde je i jeden supervýkonný server nedostačující. Každý dnešní projekt si tyto informace chrání, jelikož se jedná o jejich know-how. Osobně mám však zkušenosti s několika většími projekty. Nejčastějším řešením jsou dva servery, kde na jednom běží samotná aplikace – web a konvertování. Druhý server pak slouží jako datové úložiště a streamovací (přehrávací)

server. V extrémnějších případech je další server vyhrazen jako databázový a další server (servery) pro konvertování videa. Samozřejmě i datových serverů může být více. Tyto servery jsou propojeny vnitřní intranetovou linkou a data jsou mezi nimi přenášena po vysokorychlostních linkách. Tím se zkracuje odezva a jedná se o robustní a hlavně vysoce výkonné řešení. Pro uživatele je toto všechno samozřejmě plně transparentní.

5.2 Software

Co se týče softwaru, zde je situace velice rozmanitá. Hlavním činitelem je zde zvolené řešení – jestli bude videoserver pracovat s flash, silverlight, realplayer a nebo úplně jiným přehrávačem.

5.2.1 Operační systém

Z obecného hlediska je nutný operační systém. Mám vyzkoušenou Linux (Debian, Ubuntu) a Windows (XP, Vista, 7, 2003 Server) platformu. Většina videoserverů – stejně jako webů obecně – běží pod OS Linux.

5.2.2 Základní webový server

Dalším požadavkem je webový server – např. apache, lighttpd, TomCat nebo nginx, programovací (skriptovací) jazyk – PHP, ASP, Java atd. a databázový program – MySQL, PostgreSQL, Firebird, Oracle aj.

5.2.3 Specifický software

Výše uvedené programy jsou všeobecně známé a nutné pro provozování webhostingu jako takového. Pro videoserver je však nutné mít několik dalších programů. Především je to konvertor videí. Tedy program, který dokáže převést nahrané video do formátu, s kterým si poradí webový přehrávač. Nejznámějším programem je ffMPEG, který podporuje většinu dnes používaných formátů a kodeků. Jeho obrovskou výhodou je multiplatformost – funguje pod OS Linux, Windows i MacOS [11]. Dalšími konvertovacími programy jsou např. MEncoder nebo Transcode.

Pro úspěšný běh videoserveru by výše popsané dostačovalo. Avšak pro plnou funkcionalitu projektu je potřeba doinstalovat další software. Mezi takové patří programy opravující chybějící metadata u zkonvertovaných FLV videí – FLVTool2 nebo yamdi. Příp. pluginy do webserverů umožňující „přetáčení“ videa do libovolné části bez nutnosti mít videa načtené.

5.2.4 Shrnutí

I v případě softwaru probereme konkrétní možná řešení. Operační systém je většinou volen Linux, neboť je zcela zdarma. Není však problém rozjet videoserver i na Windows platformě. V případě Linuxu je pak volen webserver Apache, na Windows pak Microsoft IIS nebo také Apache. Databázový program je na zcela na každém uživateli, co mu vyhovuje. Dnes nejpoužívanější je rozhodně MySQL. V případě konvertovacích programů je to ffMPEG. Toto řešení – ať už Linux nebo Windows platforma – je plně dostačující pro menší weby. Ale jak bylo řečeno v části hardwarových nároků, dlouhá

videa dělají problém a hodně vytěžují server. Proto se tento problém úspěšně řeší nasazením dvou webserverů. Mám zkušenosti s projekty, kde samotná aplikace jako taková funguje na webserveru Apache a videa jsou streamována z data serveru pomocí jiného – „lehkého“ – webového serveru. Ideální pro tyto potřeby je lighttpd a nebo nginx. Velikou výhodou tohoto řešení je rozložení výkonu. Na aplikačním serveru běží kompletní webserver Apache, MySQL, PHP atd. Na datových serverech funguje už pouze jenom lighttpd nebo nginx, bez podpory PHP aj.

6 Zatížení videoserveru v závislosti na počtu uživatelů, následky a jejich řešení

Každý internetový projekt spolu s přibývající popularitou a tím pádem i více návštěvníky, vyžaduje vyšší nároky na svůj provoz a výkon celého serveru.

6.1 Hlavní problémy

6.1.1 Přenos dat (traffic)

U videoserveru platí výše zmíněné pravidlo dvojnásob. Na vině je samozřejmě velikost souborů, které jsou přenášeny směrem k uživateli. U běžných webů, kde jsou nejnáročnější na přenos dat obrázky, hravě zvládá řádově stovky i tisíce návštěvníků obyčejný webhosting. Avšak pokud má průměrně velké video třeba 10 MB, pak 100 zhlédnutí takového videa generuje přenos dat (traffic) 1000 MB. A to je už poměrně veliké číslo. Pokud si navíc vezmeme v potaz, že většina návštěvníku si během jedné návštěvy prohlédne i více videí, dostáváme se k hodně vysokým číslům.

6.1.2 Vytížení procesoru

Vysoký přenos dat však není jediným problémem, který postihuje videoservery. Pokud běží projekt na jednom serveru, při větší návštěvnosti se velmi brzy stane, že přestane stíhat procesor. Kritický stav nastává v momentě, kdy se konvertuje mnoho videí. Proces převádění videí zabere velmi mnoho času procesoru a může se stát, že se nedostane na klíčové programy v systému, mezi které bezpochyby patří webserver a nebo databázový program. V takovém případě se začne hroutit celý server.

6.1.3 Nedostatek operační paměti

Výše uvedené jsou dva hlavní specifické problémy, s kterými se dříve či později setká každý úspěšnější projekt pracující s videem. Potíží však může nastat ještě více. Pokud pomineme běžně se vyskytující chyby v aplikaci nebo špatně optimalizované dotazy, dostaneme se k nedostatku operační paměti. Pro běh se většinou používá webový server apache, který není až tak ideální pro posílání objemnějších souborů směrem k uživateli. V případě, že aplikace umožňuje dlouhá videa – řádově desítky minut – apache pak spotřebuje velmi mnoho paměti a všechna volná paměť může být brzy vyčerpána.

6.2 Příklady z praxe

Osobně mám přístup ke statistikám několika větších videoserverů a čísla jsou velice zajímavá.

6.2.1 videotube.sk

Jak zobrazuje následující obrázek, tento web navštíví denně něco kolem 30 tisíc uživatelů. Poslední sloupeček udávající přenos dat je pouze pro server, na němž běží aplikace, nejsou tam tedy započítána data, která generuje přenos videí.

Den	Počet návštěv	Stránek	Hitů	Přenos dat
01 Mar 2010	31463	340980	2364549	24.73 GB
02 Mar 2010	30956	329848	2369867	25.21 GB
03 Mar 2010	30524	319530	2246000	23.82 GB
04 Mar 2010	30070	320919	2310911	24.77 GB
05 Mar 2010	29635	334895	2305647	24.32 GB
06 Mar 2010	30511	407052	2343161	23.97 GB
07 Mar 2010	34108	415921	2500464	25.29 GB
08 Mar 2010	30765	382850	2444727	25.23 GB
09 Mar 2010	31238	306447	2289072	24.28 GB
10 Mar 2010	30578	313857	2337619	25.28 GB
11 Mar 2010	30505	359770	2459228	26.07 GB
12 Mar 2010	29518	337569	2364989	25.23 GB
13 Mar 2010	31625	362838	2325665	24.32 GB
14 Mar 2010	33877	387325	2394512	24.47 GB
15 Mar 2010	30615	337167	2327509	24.14 GB
16 Mar 2010	30503	336873	2328015	24.31 GB
17 Mar 2010	29105	341105	2195545	22.88 GB
18 Mar 2010	28037	327250	2184310	23.03 GB
19 Mar 2010	27025	376436	2319272	23.90 GB
20 Mar 2010	28492	328663	2057813	21.08 GB
21 Mar 2010	31634	351629	2269602	23.18 GB
22 Mar 2010	28301	293746	2151144	22.60 GB
23 Mar 2010	28155	286955	2018817	21.28 GB
24 Mar 2010	26652	298617	2031447	21.32 GB
25 Mar 2010	27278	307424	2101981	22.08 GB
26 Mar 2010	26439	318490	2174436	22.77 GB
27 Mar 2010	29167	321990	2112248	21.97 GB
28 Mar 2010	32023	359359	2331156	23.80 GB
29 Mar 2010	29187	301265	2212715	22.78 GB
30 Mar 2010	27659	266974	2026517	20.25 GB
31 Mar 2010	29599	277696	2241032	22.75 GB
Průměr	29846.58	333917.42	2262579.68	23.58 GB
Celkem	925244	10351440	70139970	731.11 GB

Obrázek 1 – Statistika návštěvnosti videoserveru videotube.sk (březen 2010)

Bohužel statistiky pro datový server – z kterého jsou přenášena videa k uživateli – nemám k dispozici.

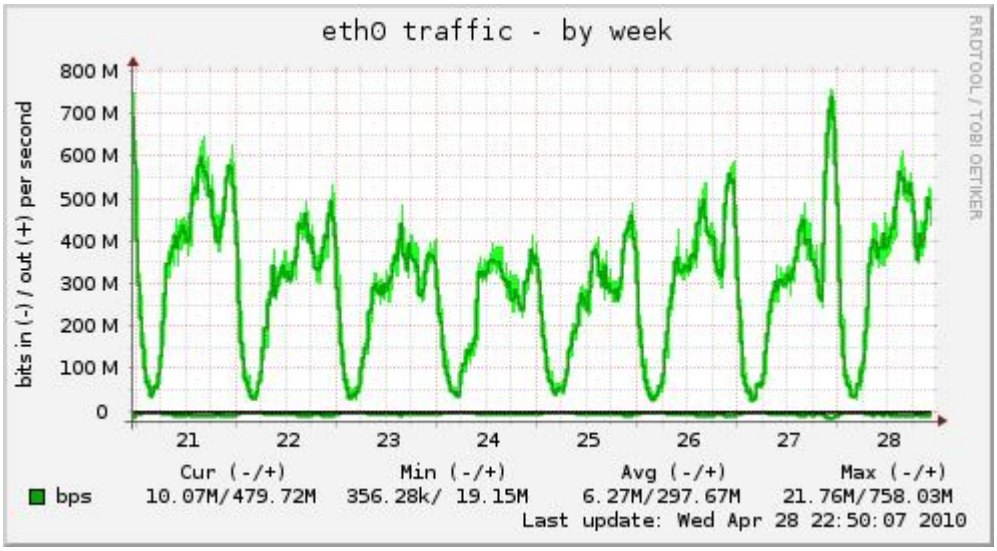
6.2.2 zvraceny.cz

Dalším videoserverem, k jehož statistikám mám přístup, je český web zvraceny.cz. Jak ukazuje obrázek, návštěvnost je zde dost nevyrovnaná a kolísá od 20 tisíc až k 50 tisíc návštěvníků denně.

Den	Počet návštěv	Stránek	Hitů	Přenos dat
01 Apr 2010	45837	739045	4328163	332.39 GB
02 Apr 2010	43867	731043	4257267	322.48 GB
03 Apr 2010	38749	636390	3711658	278.72 GB
04 Apr 2010	41876	738538	4223328	319.54 GB
05 Apr 2010	46705	831384	4799040	363.29 GB
06 Apr 2010	50984	904305	5274685	406.21 GB
07 Apr 2010	47526	806042	4740586	351.50 GB
08 Apr 2010	47871	812585	4635578	351.05 GB
09 Apr 2010	45271	747380	4347614	325.87 GB
10 Apr 2010	43269	780614	4355054	323.18 GB
11 Apr 2010	51275	947021	5249898	394.55 GB
12 Apr 2010	50900	857105	5025272	370.80 GB
13 Apr 2010	50373	856084	5002296	354.66 GB
14 Apr 2010	29311	732682	4437879	303.02 GB
15 Apr 2010	28588	686057	4314445	285.48 GB
16 Apr 2010	25830	851785	4839034	323.83 GB
17 Apr 2010	23229	732154	4189251	285.48 GB
18 Apr 2010	26968	827107	4748991	349.08 GB
19 Apr 2010	28196	868137	5003828	372.74 GB
20 Apr 2010	27803	856349	4777103	336.65 GB
21 Apr 2010	27648	900117	5021892	354.57 GB
22 Apr 2010	27754	792553	4567671	327.47 GB
23 Apr 2010	25172	742381	4299103	316.04 GB
24 Apr 2010	21956	630898	3726211	272.99 GB
25 Apr 2010	26153	694250	4191521	302.82 GB
26 Apr 2010	30317	859598	5062996	367.25 GB
27 Apr 2010	28063	750786	4370334	326.27 GB
28 Apr 2010	2244	66667	363793	26.60 GB
29 Apr 2010	0	0	0	0
30 Apr 2010	0	0	0	0
Průměr	35133.39	763537.75	4423731.82	323.02 GB
Celkem	983735	21379057	123864491	9044.51 GB

Obrázek 2 – Statistika návštěvnosti videoserveru zvraceny.cz (duben 2010)

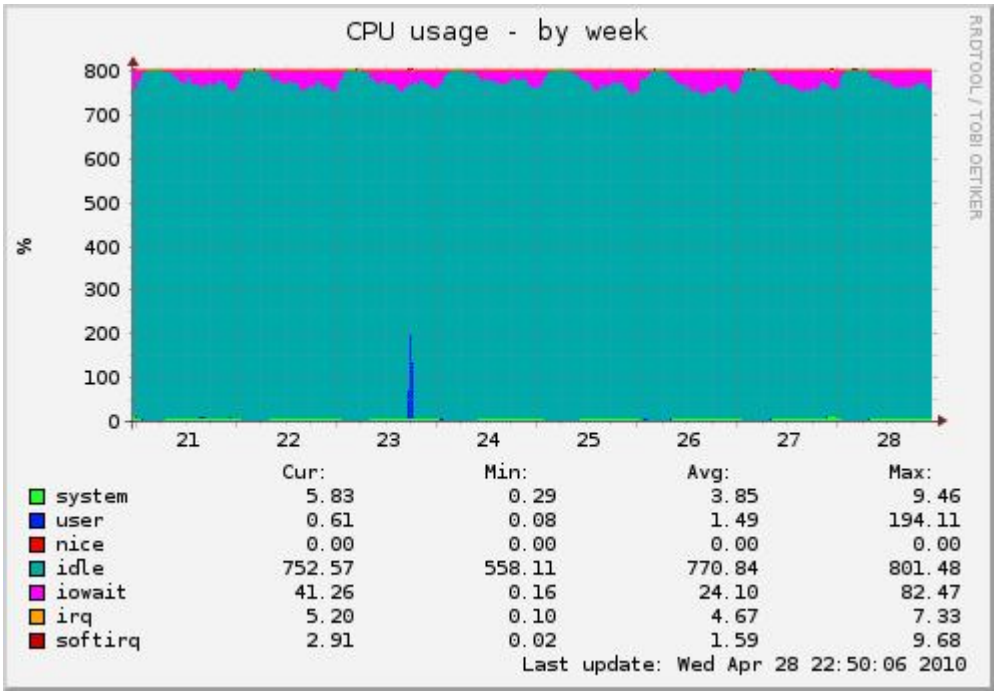
Avšak nejzajímavější statistika je počet přenesených dat. Konkrétně se jedná opět o stránku zvraceny.cz, konkrétně pak o jejich datový server, ze kterého jsou streamována videa k uživateli. Jedná se o graf průtoku dat přes síťovou kartu generovaný programem munin.



Obrázek 3 – Přenos dat na datovém serveru zvraceny.cz

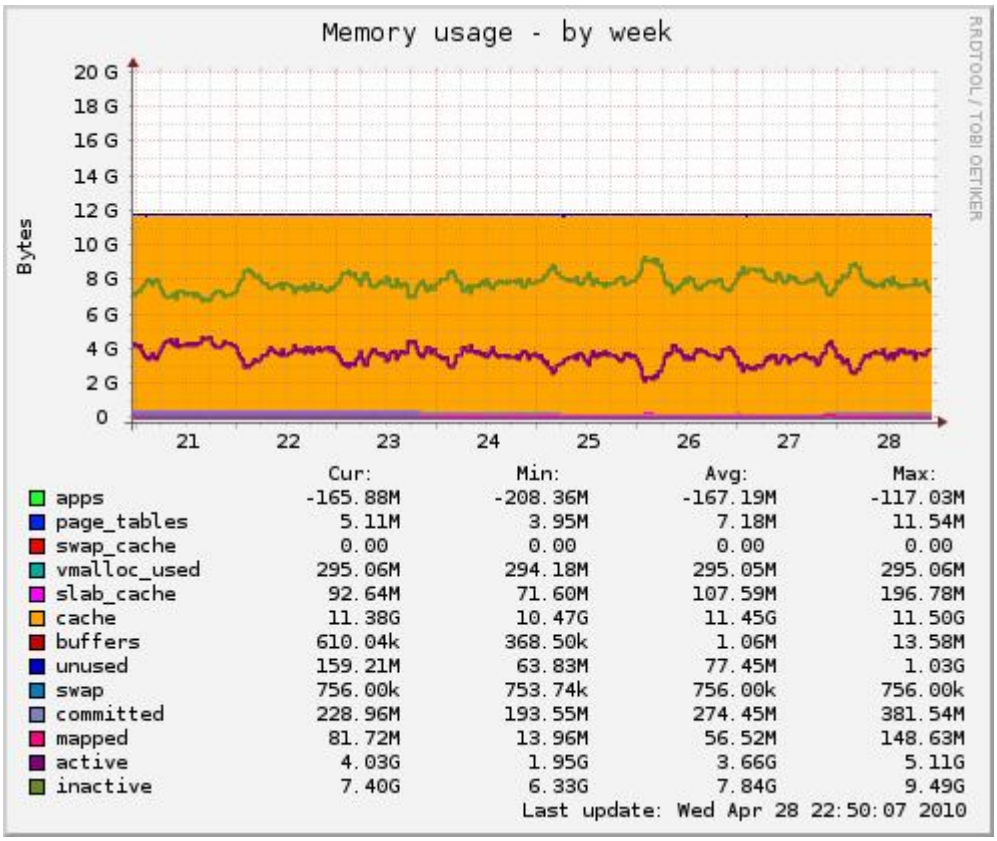
Na obrázku je patrné, že server vytváří obrovský přenos dat. Je samozřejmě znát denní doba. V noci je přenos dat na velice nízké hodnotě, ve špičce se stoupá až ke svému vrcholu. Extrémním bodem je pak čas těsně před půlnocí z 27. na 28., kdy se přenos dat blížil k hodnotě 800 Mbit/s.

Velice zajímavý graf poskytuje také vytížení procesoru. Jelikož jsou videa servírována přes webový server lighttpd, procesor je po většinu času prakticky nevytížen. Z toho je také možné si odvodit, že pro data server není potřebný až tak výkonný procesor.



Obrázek 4 – Vytížení procesoru na datovém serveru zvraceny.cz

Posledním grafem, který bych zde rád uvedl, je využití operační paměti. Díky nasazení „lehkého“ webserveru lighttpd je paměť vytěžována pouze částečně. Donedávna běžel na serveru apache a instalovaných 12 GB operační paměti bylo přesto málo. Jak je vidět na grafu, současné řešení využívá konstantně něco kolem 4 GB RAM.



Obrázek 5 – Využití operační paměti na datovém serveru zvraceny.cz

6.3 Řešení problémů

Na začátku kapitoly bylo zmíněno, že každý úspěšnější projekt se začne potýkat s několika zásadními problémy. Nyní bude vysvětleno, jak tyto problémy úspěšně řešit.

6.3.1 Přenos dat (traffic)

Asi prvním a nejčastějším problémem, který se vyskytne, je nedostačující konektivita. Jak bylo možné vidět na grafech, videoserver generuje obrovský přenos dat. Běžně dostupná konektivita u českých hostingových firem je 100 Mbit/s. To je však hodnota, která ve většině případů nedostačuje. Především dnes, v době rychlých internetových připojení domácností, je to problém o to závažnější.

Pokud by se sešlo v jednu chvíli na webu 12 lidí s rychlostí připojení 8 Mbit/s, konektivita serveru by byla prakticky vyčerpána. Nejradikálnějším řešením je samozřejmě navýšit konektivitu serveru. Optické přípojky dnes nejsou problém, jenže konektivita je drahá záležitost, zvlášť pokud se jedná o tu zahraniční, kde se za každý další Mbit/s platí i 10 €. Proto je nutné hledat jiná řešení a projekt optimalizovat.

Asi nejlepším se jeví být jisté omezení přístupu k videím. Především se jedná o snížení rychlosti stahování dat. Většina webserverů umožňuje přímo v nastavení určit maximální rychlost přenosu dat. Jedná se o velice elegantní a hlavně účinné řešení. Pokud se zvolí vhodná rychlost, uživatel tuto změnu ani nepostřehne. Důležité je však ponechat hodnotu a něco vyšší, než je datový tok videa. Pokud by byl průměrný datový tok videa 500 kbit/s a maximální rychlost bychom nastavili třeba na 300 kbit/s, pak by se video v jednom kuse „trhalo“.

Dále je možné omezit maximální počet připojení jednoho uživatele na server. Jednoduše řečeno – aby jeden uživatel nestahoval (nepřehrával) v jednu chvíli třeba 10 videí a nevytěžoval tím zbytečně linku.

Určitým odlehčením pro konektivitu serveru je zpřístupnění tzv. „seekování“. Jedná se o funkci videa a přehrávače, kdy uživatel může kliknout na libovolné místo na časové ose a aniž by bylo video kompletně načteno, začne se přehrávat od zvoleného místa. Velmi mnoho lidí si video jen tak „prohlíží“ a „proklikává“, nedívá se na něj celé. A pokud je ono seekování dostupné, návštěvník si klikne, kam potřebuje a výrazně se tím ušetří konektivita – video se nestahuje celé, jen ta část, kterou uživatel žádá.

6.3.2 Vytížení procesoru

Procesor je samotným procesem konvertování velice vytížen a velmi rychle se může stát, že přestane reagovat v dostatečném čase a v krajních případech se může stát, že se celý server zhroutí. Lze tomu však do jisté míry předejít.

Podobně jako u problémů s konektivitou lze i toto vyřešit hardwarově. V tomto případě to znamená lepší a výkonnější procesor. To jsou však další investice, mnohem účinnější cestou je optimalizace.

Už při prvotním návrhu aplikace je nutné si důkladně promyslet, který formát a kodek bude projekt využívat. Je hezké, když si vybereme vysoce kvalitní H.264 kodek s profilem maximální kvality. Výsledné video bude dosahovat úžasné kvality, jenže proces konvertování videa se může také protáhnout do řádů hodin. Naproti tomu formát FLV, který nepatří svojí kvalitou mezi špičku, je na tom rychlostně skvěle. Konkrétní porovnání těchto dvou formátů je obsahem jiné kapitoly, spokojme se teď s faktem, že rozdílná rychlost konvertování může být až v desetinásobcích. Proto je vhodné volit jakýsi kompromis mezi kvalitou videa a rychlostí zpracování.

Každý projekt má k dispozici nějaká statistická data. A pokud víme, že se na našem webu denně nahraje maximálně 20 videí, nemá asi zatím cenu přemýšlet nad optimalizací. Avšak pokud by počet nahraných souborů stoupal a míjel hranici stovek nahrání denně, pak by bylo vhodné se nad nějakou optimalizací zamyslet.

Největší potíže budou zřejmě nastávat během špičky, kdy bude na webu nejvíce návštěvníků, tím pádem bude přibývat nejvíce videí a také bude největší zátěž na server. Pro tento případ se velice hodí nasazení konvertovací fronty. Princip je jednoduchý.

Uživatel nahraje na web video, avšak video se hned nezačne konvertovat, nýbrž se vloží do fronty. Z této fronty se budou postupně vybírat videa podle pořadí a konvertování bude probíhat postupně. Výhodou je zde samozřejmě odlehčení zátěže – v jednu chvíli bude probíhat pouze vždy jen jedno konvertování. Avšak dost zásadní nevýhodou je prodloužení doby čekání uživatele od nahrání svého videa.

Když není fronta implementována, video se začne převádět ihned po nahrání. V případě nasazení fronty se však na řadu může dostat až třeba i za několik hodin. Tento problém lze do jisté míry eliminovat nasazením více front. V jednu chvíli tedy bude probíhat n konvertování, kde „n“ je rovno počtu front. Ideální počet těchto front závisí na daném prostředí a nelze obecně určit vhodnou hodnotu. Je to víceméně metoda pokus-omyl a sledováním statistik a vytížení serveru.

Ve velmi extrémních případech, kdy jsou na web nahrávány řádově stovky nových videí denně a procesor nestíhá obsluhovat všechny služby (webserver, databáze atd.) a do toho konvertování videí, lze vyhradit speciální server pouze pro převod. Takový server má pak velice specifickou hardwarovou konfiguraci – vysoce výkonný procesor, naproti tomu není potřeba moc diskového prostoru. Video je „dopraveno“ na tento speciální server, převedeno do požadovaného formátu a opět putuje dál – na datový server.

6.3.3 Nedostatek operační paměti

Většina programátorů a lidí obecně si neradi komplikují život, a proto dělají všechno tzv. na „jedné hromádce“. V tomto případě je myšlen webserver. Převážná část vývojářů naprogramuje aplikaci, která pak běží na jejich oblíbeném webovém serveru, stejně tak streamování videa. Bohužel to není až tak optimální řešení. Webserver apache je vynikající program se spousty nastavení, není však úplně ideální pro „servírování“ objemnějších souborů, mezi které video rozhodně patří. V takovýchto případech pak velice rychle dochází operační paměť na serveru a také procesor je zatížen víc, než by bylo nutné.

Ideálním řešením je v tomto případě nasadit nějaký „lehký“ webový server. Osobně mám velice dobré zkušenosti s lighttpd a nginx. Aplikace zůstane nadále nezměněna a dál poběží na apache. Změna nastane u souborů s daty. Na server se nasadí nějaký „lehký“ webserver, který využívá úplné minimum systémových prostředků a je proto vhodným kandidátem pro streamování velkého množství videosouborů.

7 Vlastní řešení videoserveru

Ze softwaru jsem si pro svůj projekt vybral operační systém Linux, konkrétně distribuci Debian. Důvod výběru OS Linux byl prostý – stabilita a cena. Linuxové servery jsou ověřené léty zkušeností a navíc jsou zcela zdarma ke stažení, distribuci Debian pak proto, že je mi práce s ním nejbližší. Skriptovací jazyk jsem zvolil PHP, neboť s tímto skriptovacím jazykem už léta pracuji a mám s ním veliké zkušenosti. Zároveň je také šířen pod velice volnou licencí – opět zcela zdarma. Na databázový program padla volba na MySQL. Stejně jako v PHP i s touto databází mám dlouholeté zkušenosti, opět je zde svobodná licence. Konvertovaní software jsem si vybral FFmpeg, v minulosti jsem s tímto programem již pracoval a osobně se mi zdá jako nejlepší variantou. Funguje na Windows, Linuxu i MacOS a je opět volně ke stažení. Přehrávač videí jsem zvolil JW Flash Video Player. Jak název napovídá, jedná se o přehrávač pro Adobe Flash. Je to velice výkonný přehrávač se spoustou možností nastavení, množstvím skinů atd.

7.1 Volba formátu videa

V praktické práci jsou používány dvě kvality videa (viz níže), výběr formátu padl na klasický FLV kontejner s kodekem VP6 pro nižší kvalitu, pro video ve vysoké kvalitě je pak použit MP4 kontejner s H.264 kodekem.

7.1.1 FLV

FLV je zkratka pro „Flash Video“ a jedná se o formát multimediálního kontejneru. V současné době se používá především v přehrávačích Adobe Flash. Podpora tohoto kontejneru začala ve verzi Adobe Flash Player 6. Videokodeky, pomocí kterých mohou být data kódována, jsou Sorenson Spark (Soronsen h.263), VP6, Screen video, v posledních verzích je nově přidána také podpora H.264. Co se zvukových formátů týče, podporovány jsou MP3, AAC, ADPCM, Linear PCM, Nellymoser, Speex a G.711 [11].

Formát FLV je všeobecně známý jako soubor s příponou *.flv. Unixové systémy používají tzv. MIME typy, zde je FLV prezentován jako video/x-flv. V jednom souboru nemůže být víc než jedna audio a jedna video stopa.

Obsah FLV souboru začíná vždy hlavičkou. První tři znaky souboru jsou identifikační a obsahují znaky F (0x46), L (0x4C) a V (0x56). V hlavičce dále najdeme informace o verzi FLV, informace o zvuku a videu a také pozici startu těla souboru – jinými slovy velikost hlavičky. Samotné tělo souboru je tvořeno střídavě dvěmi informacemi

1. velikostí předchozího tagu
2. tag (viz dále)

Každý tag obsahuje informace, jestli se jedná o audio, video nebo jiná data, dále je to velikost (délka) samotných dat, čas (timestamp) v milisekundách, který je relativní

k prvnímu tagu (první tag má timestamp = 0). Každý tag je očíslován pomocí tzv. StreamID.

Zde přichází na řadu další dělení, a to konkrétně, zda jsou data video, zvuk a nebo jiná data. V případě audia jsou v každém tagu samotná data, plus informace popisující zvuk samotný – konkrétně formát zvuku (MP3, AAC atd.), frekvence zvuku (44 kHz, 22 kHz atd.), bitová hloubka (8 nebo 16 bit) a typ zvuku (mono nebo stereo). U videa jsou to také samotná data a informace popisující video – použitý kodek a typ snímku (např. klíčový snímek). FLV soubor může také obsahovat speciální metada, která jsou volána pomocí metody onMetaData ve Flash přehrávači. Jedná se např. o informace, kdo video vytvořil, o délce (trvání), velikosti audio a video stopy, rozlišení videa (výška, šířka), snímků za sekund (fps) a především obsahuje pozice klíčových snímků [13]. Proč je tato informace tak důležitá bude vysvětleno dále.

7.1.2 VP6

TrueMotion VP6 je videokodek vytvořený společností On2 Technologies jako nástupce úspěšných předchůdců VP3 a VP5. Nejčastěji je využíván v Adobe Flash a v JavaFX. VP6 byl představen v květnu 2003, o čtyři měsíce později (září 2003) byl oficiálně vydán pod názvem TrueMotion VP6. V listopadu 2003 byl tento kodek zvolen pro China's Enhanced Versatile Disc (EVD), což je konkurent formátu DVD. Po rozporech mezi firmami a vládou byl však vývoj formátu EVD zastaven [14]. V současné době se jedná už do jisté míry za zastaralý kodek a byl kvalitou překonán. Jeho velikou výhodou je však nenáročnost na procesor – video se plynule přehrává i na starších počítačích, zároveň rychlost konvertování je velice rychlá.

7.1.3 MP4

MP4 (MPEG-4 Part 14), formálně ISO/IEC 14496-14:2003 je formát multimediálního kontejneru specifikovaný ve standardu MPEG-4. Nejčastěji se používá pro audio a video, je však schopen nést informace o titulcích a také obrázcích. Jako jeden z nejmodernějších kontejnerů umožňuje streamování přes internet. Oficiální označení přípony je *.mp4, MIME typ pak jako video/mp4, audio/mp4 a nebo také application/mp4.

Tento formát vychází z kontejneru QuickTime od společnosti Apple, první verze MP4 byla publikována v roce 2001. Tato verze byla roku 2003 nahrazena verzí 2, konkrétně specifikací MPEG-4 part 14.

MP4 kontejner může nést video s kodeky MPEG-4 Part 10 (též označován jako H.264/MPEG-4 AVC), MPEG-4 Part 2 a nebo méně používané MPEG-2 nebo MPEG-1. Z audiokodeků je pak možné použít AAC (MPEG-4 Part 3 Subpart 4), MP3, MP2, MP1, CELP, HVXC a další. Jak bylo zmíněno, tento kontejner může obsahovat i titulky. Podporovány jsou MPEG-4 Timed Text (známé pod označením 3GPP Timed Text), někdy jsou používány Vobsub [15].

7.1.4 H.264

H.264/AVC/MPEG-4 Part 10 je standard pro kompresi videa, jehož první verze byla zveřejněna v květnu 2003. Jedná se o jeden z nejpoužívanějších kodeků současnosti. Zvlášť poslední dobou, kdy se čím dál více klade důraz na kvalitu, jeho popularita roste. Je využíván v Blu-Ray, jsou v něm konvertována videa na youtube, iTunes, používá se pro videokonference. A aniž bychom si to uvědomovali, tímto kodekem je kódována převážná část digitálního vysílaní, satelitních přenosů, kabelových televizí atd. Velmi často je tento kodek využíván u HD kamer.

Od začátku vývoje byl kladen důraz na vysoký kompresní poměr a výborného obrazu v nižších datových tocích než předchozí formáty (např. polovina nebo méně než MPEG-2) [16]. Tento záměr se povedl, kodek dosahuje vynikajících výsledků – video kódované pomocí H.264 má výbornou kvalitu obrazu a zároveň přijatelnou velikost výsledného souboru. Vykoupením za tyto výsledky je mnohem vyšší vytížení procesoru, než je tomu třeba u FLV. Na méně výkonných sestavách bude problém takové video vůbec přehrát. A samotné kódování je také mnohonásobně delší než u formátu FLV. Podrobně se tomuto problému budeme věnovat v další části.

7.2 Volba audio formátu

Jak bylo řečeno u výběru video formátů, v praktické části jsou dvě kvality videa. Stejně tak je tomu i u zvuku. U nižší kvality (video FLV) byl vybrán velice rozšířený formát MP3. V případě vyšší kvality (video MP4) padla volba na audio formát AAC.

7.2.1 MP3

MP3 (MPEG-1 Layer 3) je ztrátový formát zvuku, který umožňuje zachovat vysokou kvalitu zvuku, přičemž velikost výsledného souboru je výrazně menší než originál. Princip fungování je takový, že se ze zdrojového zvuku odeberou informace, jež člověk neslyší a nebo je nevnímá. Tím je odstraněno velké množství dat při zachování solidní kvality. Při vyšších datových tocích (256 kbit/s a výše) je rozdíl od originálního CD pro normálního člověka prakticky neslyšitelný. Nevýhodou je horší kvalita mluveného slova. Tento formát se stal zvukovým standardem a je všeobecně v povědomí lidí – laiků [17].

7.2.2 AAC

Advanced Audio Coding (AAC) je, podobně jako mp3, ztrátový formát zvuku. Byl vyvinut jako pokračovatel formátu MP3 pro střední a vyšší datové toky. Kvalita tohoto formátu je však velmi nestálá, neboť přímo závisí na použitých kodecích, které jsou velmi nevyrovnané. Na jedné straně prvotřídní AAC enkodér od Apple, na straně druhé ne příliš vyvedený FAAC. V povědomí veřejnosti není příliš znám, v oblasti hardwaru se ujal především v přehrávačích Apple iPod a Sony Play Station Portable [18].

7.3 Srovnání kvality videokodeků

Volba formátů videa (FLV – VP6 a MP4 – H.264) nebyla náhodná. Současné tendence směřují spíše ke kvalitě. Hardwarový výkon dnes předčí softwarové nároky, tudíž není problém ani po této stránce. Jak bylo řečeno v popisu FLV formátu a VP6 kodeku, nejedná se o nejlepší možnou kvalitu. Z hlediska praktického nasazení by bylo zřejmě vhodnější nasadit i pro nižší kvalitu kodek H.264, jelikož se jedná o současnou špičku. Má volba pro FLV (VP6) padla právě pro srovnání kvality těchto dvou formátů.

Velice důležitou poznámkou, kterou je třeba zmínit, je fakt, že nejnovější verze programu FFmpeg má problémy s formátem FLV. Konkrétně tento konvertovací program ignoruje parametr –b, který určuje datový tok videa. Není to však zásadní problém, výsledné video má datový tok kolem 2000 kbit/s. Přímé srovnání s kodekem H.264 je tedy bez problému možné.

Jako zdrojový soubor posloužil krátký klip s trváním zhruba 4 a půl minuty, rozlišením 1280x720 bodů, datovým tokem cca 2300 kbit/s, kodekem H.264 v kontejneru MP4. Jak naznačují parametry, jedná se o kvalitní video ve vysoké kvalitě. Z každého zde testovaného vzorku byl pomocí programu FFmpeg vytvořen jeden snímek ve formátu PNG. Úmyslně byla volena doba videa, kdy se mořská voda tříští o skály. V té době je velice rychlá změna scény a jsou zde krásně vidět detaily (nedokonalosti) jednotlivých kodeků. Zároveň je ve videu text, na kterém jsou též velice snadno pozorovatelné chyby.

Obrázek 6 - Originální video

Na obrázku z originálního videa lze vidět, že se jedná o kvalitní zdroj. Text je zcela ostrý, bez výraznějších skvrn, stejně tak detail skály v pravé části obrázku. Tříštící se voda je také velice ostrá. Menší nedostatky jsou na obloze – barvy nejsou příliš věrné.

7.3.1 Formát FLV s kodekem VP6

První na řadu přisel původní formát využívaný v Adobe Flash, tedy kodek VP6 v kontejneru FLV.

Obrázek 7 – FLV s kodekem VP6 a datový tok 2000 kbit/s

Formát FLV s kodekem VP6 velmi pokulhává svoji kvalitou. Již na první pohled je znát celková neostrost obrázku a pro VP6 tak klasický efekt kostičkování obrazu. Nejzásadnější změny oproti originálu jsou patrné především v neostrosti textu a u tříštící se vody. Video s touto kvalitou by bylo poměrně bez problému sledovatelné v nějakém menším okně. Pokud by si však uživatel přepnul video do celoobrazového režimu, byl by zděšen. Toto video je pro zhlížení na celé obrazovce prakticky nekoukatelné.

7.3.2 H.264 v kontejneru MP4

Situace u kodeku H.264 je velice zajímavá, neboť zde datový tok není až tak primárním činitelem kvality. V programu FFmpeg jsou speciálně pro tento kodek dostupné tzv. profily. Jedná se o textové soubory ve formátu libx264-XXX.ffpreset, kde XXX značí název profilu. V samotném FFmpeg je pak profil aktivován pomocí parametru –pre. Ihned po instalaci je k dispozici mnoho profilů, např. libx264-default.ffpreset, libx264-fast.ffpreset, libx264-fastfirstpass.ffpreset, libx264-hq.ffpreset, libx264-ipod640.ffpreset, libx264-lossless_fast.ffpreset, libx264-main.ffpreset, libx264-max.ffpreset, libx264-medium.ffpreset, libx264-normal.ffpreset nebo libx264-ultrafast.ffpreset.

Na první pohled je hned z názvu možné určit, co který profil dělá. V zásadě platí jedno pravidlo – čím je samotný proces konvertování rychlejší, tím je horší kvalita. Pokud bychom měli být konkrétní, takový profil s názvem ultrafast bude velice rychle

konvertovat, ale výsledná kvalita videa nebude nic moc. Naproti tomu profil s označením hq bude sice pomalý, ale kvalita obrazu bude velice dobrá.

Obrázek 8 – MP4, kodek H.264, datový tok 500 kbit/s, profil max

Z obrázku je značně patrné, že byl zvolen příliš nízký datový tok a ačkoliv byl zvolen nejlepší profil (max), ani to videu nepomohlo. Obraz je velice rozostřený, pro sledování v celoobrázkovém režimu prakticky nepoužitelné. Na druhou stranu je třeba uznat, že video není o tolik horší, než v případě VP6 kodeku, kde byl však datový tok čtyřnásobný (2000 kbit/s).

Obrázek 9 – Kodek H.264 v MP4 kotejneru, datový tok 2000 kbit/s, profil ultrafast

Na předchozím snímku jsou potvrzena výše uvedená slova – nezáleží pouze na datovém toku. Ačkoliv je datový tok 2000 kbit/s, což se rovná skoro originálu, obraz je velice nekvalitní, neostrý a kostičkovaný. Důvodem je použití nejrychlejšího konvertovacího profilu – ultrafast.

Zveřejňování dalších obrázků by bylo zbytečné, testy byly prováděny také pro profily default a normal. Asi není třeba dodávat, že u profilu max a bitrate 2000 kbit/s byl originál prakticky nerozeznatelný.

7.3.3 Časy konvertování

Výsledek obrazové kvality je zcela zřejmý. Na plné čáře vítězí H.264 s profilem max. Kvalita této volby je nekompromisní, video lze sledovat bez problému v celoobrázkovém režimu a i v případě konvertování z vysoce kvalitního zdroje (např. FullHD s bitrate převyšující 10 Mbit/s) je výsledné video velmi obstojné vůči originálu.

Problémem je však čas, konkrétně doba samotného konvertování. Bylo prováděno několik měření, výsledné hodnoty pak uvádí následující tabulka. Pro úplnost, videa byla konvertována v OS Windows 7 64bit programem FFmpeg s plně nevytíženým systémem. Testovací sestava obsahovala dvoujádrový procesor AMD Athlon X2 4800+ (2x 2,4 GHz) a 2 GB RAM.

Kontejner	Kodek	Datový tok	Profil	Čas konvertování (min)
FLV	VP6	2000 kbit/s	---	3:51
MP4	H.264	500 kbit/s	default	4:49
MP4	H.264	500 kbit/s	normal	6:18
MP4	H.264	500 kbit/s	ultrafast	3:43
MP4	H.264	500 kbit/s	max	22:04
MP4	H.264	2000 kbit/s	default	6:20
MP4	H.264	2000 kbit/s	normal	8:41
MP4	H.264	2000 kbit/s	ultrafast	3:59
MP4	H.264	2000 kbit/s	max	32:16

Tabulka 1 – Časy konvertování

Z tabulky jasně vyplývá, že čím kvalitnější profil je použit, tím delší dobu konvertovací proces trvá. Je proto nutné zvolit nějaký kompromis mezi kvalitou obrazu a časem převodu.

7.4 Popis webové stránky videoserveru

7.4.1 Layout, rozvržení a grafika

Při návrhu internetových stránek byl po všech směrech kladen důraz především na jednoduchost a přehlednost. V horní části nalezneme logo, pod ním pak tři hlavní „záložky“. Ty slouží pro přepínání mezi hlavními částmi webu. Úplně první záložkou jsou „Videa“. Přes tento odkaz se dostaneme na úvodní stránku – tedy seznam videí. Pomocí odkazu „Kategorie“ si budeme moct prohlížet seznam kategorií. Po kliknutí na kategorii se zobrazí výpis videí zařazených do dané kategorie. Poslední záložkou je pak „Nahrát video“, což umožňuje uživatelům nahrát nový soubor na web. Pod záložkami najdeme hlavní obsahovou část, jejíž obsah závisí na aktuální stránce. Po pravé straně pak najdeme formulář pro přihlášení uživatele, resp. pokud je uživatel již přihlášen, informace o přihlášení – jméno, počet videí a poslední přihlášení. V pravé části (pod přihlašovacím formulářem) se též nachází pole pro vyhledávání. Stránku uzavírá podpis autora v pravém dolním rohu.

Grafická úprava je směřována k dnešním moderním internetovým stránkám. Web dodržuje webové standardy, konkrétně se jedná o HTML 4.01 Transitional a CSS 2.1.

V rozsahu celých stránek jsou používány základní techniky SEO optimalizace pro lepší umístění ve vyhledávačích. Využíván je modul mod_rewrite, který zajišťuje hezčí adresy. Z nepěkného www.neco.cz/index.php?stranka=kategorie&ID=23 tvaru je pak možné udělat o mnoho srozumitelnější adresu, např. www.neco.cz/kategorie/23-auto-moto/.

Při tvorbě stránek bylo pracováno se standardními funkcemi skriptovacího jazyka PHP, není tedy použit žádny framework (Zend, nette atd.). Pro přístup k databázi jsou využívány vlastní funkce vycházející ze základních funkcí jazyka PHP. Opět tedy není využito žádných tříd třetích stran. Výjimkou je použití knihovny ulehčující práci s JavaScriptem – jQuery. Pro práci s obrázky (změna velikosti) je použita externí třída dostupná na http://www.mawhorter.net/web-development/simple-image-manipulation-in-php-rotate-resize-crop-flip-and-mirror-thumbnails-square-and-regular.

Výpis seznamu videí je vždy jednotného stylu. Tento výpis se používá

- na hlavní stránce (záložka Videa)
- ve výpisu videí v kategorii
- vyhledávání

Videa jsou v tomto výpise přehledně řazena pod sebou, u každého videa nechybí náhled a základní informace – název, popis, grafické hodnocení (hvězdičky), datum vložení, počet zobrazení, počet komentářů a kategorie, ve které je video umístěno. Pokud je video dostupné i ve vysoké kvalitě, je náhled videa v pravém dolním rohu opatřen značkou „HD“. Uživatel tak na první pohled vidí, že je dostupná i kvalitnější varianta videa. Všechny výpisy videí umožňují řazení podle mnoha kritérií. Standardně jsou videa řazena podle data nahrání, je možné si dále zvolit řazení podle hodnocení, počtu komentářů a také podle počtu zobrazení. Vždy pak volitelně vzestupně a nebo sestupně. Každý seznam je dělen na stránky z důvodu lepší přehlednosti. Na jedné straně nalezneme vždy maximálně 10 videí. Stránkování je umístěno nad i pod seznamem videí.

Po kliknutí na název videa se dostaneme na samotný detail videa a máme možnost jej přehrát. Na stránce nalezneme opět základní informace – tedy název, popis atd., podobně jako u seznamu videí. Navíc je zde však možnost video ohodnotit a komentovat s ostatními. Je tedy možné zanechat vzkaz přímo autorovi videa a nebo komukoliv dalšímu. Jedná se o velice rychlou a hlavně účinnou zpětnou vazbu pro uživatele, jenž video nahrál. Hodnocení a komentáře jsou řešeny technikou zvanou AJAX, což jsou speciální metody jazyka JavaScript, kdy se načítají informace na pozadí. Obrovskou výhodou toho řešení je, že se nemusí znova načítat stránka, vše se děje na pozadí. Pokud si tedy uživatel právě přehrává video a klikne na hodnocení, video se bez problému dál přehrává a hlas se úspěšně započítá – bez nutnosti znovu načíst stránku. Dominantním prvkem je pak samozřejmě samotný Flash přehrávač, kde lze požadované video zhlédnout. Samotnému přehrávači se budeme věnovat v další části.

Rozhodně nejsložitější částí videoserveru je nahrávání videí. Po kliknutí na záložku „Nahrát video“ se uživatel dostane k formuláři, kde vybere soubor ze svého počítače, který chce nahrát, vyplní název, popis, klíčová slova a vybere kategorii, do které chce svoje video zařadit. Všechny zadávané hodnoty jsou dynamicky kontrolovány pomocí JavaScriptu a uživatel je ihned upozorněn v případě, kdy vyplnil nějaké pole špatně, popř. vůbec. Po úspěšné kontrole vyplněných údajů se začne soubor nahrávat na server. Stav uploadu může uživatel sám kontrolovat. Je použito rozšíření serveru PECL uploadprogress, které umožňuje sledovat stav nahrávání.

Jelikož mohou nahrávat videa pouze přihlášení uživatelé, stejně tak jako vkládat komentáře, je nutná registrace uživatelů. Registrace je řešena pouze na základní úrovni, aby návštěvníky příliš nezdržovala. V registračním formuláři je proto nutné vyplnit pouze uživatelské jméno, platnou e-mailovou adresu, dvakrát zadat požadované heslo a souhlasit s podmínkami. Podobně jako je tomu u sekce s nahráváním videa, i zde jsou všechna pole sledována pomocí JavaScriptu. Ještě před samotným odesláním registračního formuláře je tedy zkontrolován formát e-mailové adresy, formát jména a shoda hesel. Samotný proces registrace samozřejmě myslí na duplicitu, a tak následný skript kontroluje, jestli už neexistuje v systému stejné jméno a nebo stejná e-mailová adresa. Pokud všechno proběhne v pořádku, uživatel je úspěšně zaregistrován. Poté mu přijde e-mail, kde musí svůj účet aktivovat. Po aktivaci má již plně funkční účet.

7.4.2 JW Player

S více než jedním milionem aktivním uživatelů je JW Player jedním z nejvíce oblíbených a open-source multimediálních přehrávačů. Podporuje přehrávání všech formátů pro Adobe Flash – FLV, MP4, MP3, AAC, JPG, PNG a GIF. Zároveň podporuje streamováni z http a RTMP, živé vysílání, možnost modifikace pomocí spousty vzhledů (skinů), velice podrobného nastavení a také mnoho dostupných rozšíření (pluginů) [19].

Použití přehrávače je velice snadné, napomáhá tomu průvodce umístěný na oficiálních stránkách prohlížeče, kde si požadované nastavení každý doslova „nakliká“ a nechá si jej vygenerovat přímo průvodcem a vloží na svoje webové stránky. Standardně je využíváno JavaScriptové knihovny swfObject, která velice elegantně vloží swf (aplikace pro Adobe Flash) na webové stránky. Odpadá pak nutnost řešit podporu pro různé prohlížeče atd. Naprosto elementární kód, který zobrazí JW Player na stránkách, je uveden v příloze A této práce.

Obrázek 10 – JW Flash Player 5.1

Obrázek JW Flash Playeru jasně ukazuje, že se jedná o velice přehledný přehrávač. Oficiálně je sice vytvářen v angličtině, ale každý pochopí, co které tlačítko dělá, neboť je vše poznat podle grafiky.

Velikou předností tohoto přehrávače je možnost použití spousty doplňků (pluginů). Tyto pluginy není třeba nijak stahovat a umisťovat na web, neboť si je přehrávač sám stáhne z internetu a programátor se nemusí o víc starat. Mnoho doplňků vydává přímo firma Longtail, která je za vývoj JW Playeru zodpovědná. Jelikož se jedná o open-source přehrávač, samotná komunita kolem toho projektu vytvořila stovky dalších více či méně povedených pluginů. I v této práci je využíván jeden plugin. Konkrétně se jedná se doplněk „HD“, který dovoluje přepnout video do vyšší (HD) kvality. Ve zdrojovém kódu se pak pouze nadefinují dvě cesty k videu (klasická + HD) a po stisknutí tlačítka „HD“ se buď aktivuje a nebo deaktivuje HD režim. Seznam některých dalších zajímavých pluginů [20]

- Google Analytics – jak název napovídá, jedná se napojení do statistik společnosti Google. Umožňuje sledovat nejen počet zobrazení přehrávače, ale také mnoho událostí – např. zastavení videa atd.

- Facebook It – velice jednoduchý doplněk umožňující sdílení videa na sociální síti Facebook.

- Tweed It – prakticky to samé jako Facebook It, v tomto případě podporuje sdílení na Twitteru.

- HD – o tomto doplňku již byla řeč a je využíván v tomto projektu pro přepínání zobrazované kvality videa.

- Revolt – přidá to přehrávače tzv. vizualizace, známé z hudebních přehrávačů Windows Media Player, Winamp apod.
- Captions – velice zajímavý a užitečný plugin, který přidá do obrazu titulky. Ty lze v případě potřeby samozřejmě uživatelem vypnout.
- TipJar – pokud provozujeme nějaký web a chceme návštěvníky poprosit o finanční příspěvek, je tento plugin právě pro nás. Do obrazu umístí tlačítko „Donate“, po jehož stlačení dojde k přesměrování na platební systém PayPal a majiteli stránek může uživatel poslat finanční příspěvek.
- Screenshoot – s tímto doplňkem může uživatel vytvořit tzv. screenshot (obrázek z videa) a uložit si jej ve formátu JPEG.

7.4.3 Databáze

Koncept webové stránky je řešen tak, aby byl nezávislý na typu databáze. Jak bylo uvedeno výše, jsou používány vlastní funkce pro práci s databází. Pokud bude soubor s těmito funkcemi nahrazen, není problém ihned přejít na jiný databázový software. V této práci je používáno MySQL, jelikož se jedná o jednu z nejvíce oblíbených databází. Celá databáze je řešena co nejjednodušším způsobem, dotazy nejsou nikde příliš složité, je striktně dbán důraz na vysoký výkon aplikace. Všechny vstupy jsou kontrolovány pomocí regulárních výrazů, popř. ošetřeny funkcí addslashes, to vše proti možnému útoku přes SQL injection. Aplikace by měla být tedy i po této stránce robustní.

Pojmenování tabulek i všech sloupců v nich, je voleno vhodně tak, aby bylo na první pohled jasné, co má jakou funkci. Tabulek je pouze pět

- kategorie
- klicova_slova
- komentare
- uzivatele
- videa

Už z názvů je hned jasné, co se ve které tabulce nachází, proto se nebudeme podrobným popisem tabulek zabývat.

Jeden dotaz je však v aplikaci velice zajímavý. Jedná se o dotaz používaný při vyhledávání videí. Tento dotaz testuje shodu hledaného slova v názvu, popisu a klíčových slovech. Zároveň je využíváno fulltextové vyhledávání, je tedy možné použít operátory *, - atd. Dotaz je možné zhlédnout v příloze B.

7.5 Proces konvertování videa

Jakmile uživatel na stránce s formulářem pro nahrání nového videa vyplní potřebné údaje a soubor se úspěšně nahraje na server, probíhá nejdůležitější a nejnáročnější proces celého videoserveru – proces konvertování.

Samotné řešení v tomto projektu je velice sofistikované. Jelikož je převádění videa velice náročné a jak ukázaly naměřené hodnoty u kvality videa také velice časově zdlouhavé, vše je prováděno na pozadí.

Pro uživatele je postup následující

1. vyplní potřebné údaje a nahraje soubor na server
2. po dokončení nahrání je přesměrován na stránku s informací, že je video konvertováno
3. až se video na serveru zkonvertuje, je uživateli poslán email s informací o dokončení

7.5.1 Zpracování nahraného souboru

V první řadě je nutné zpracovat soubor, který uživatel nahrál, zároveň tak uložit údaje, které vyplnil – název, popis, klíčová slova atd.

Skript, který se o toto stará se jmenuje „nahrat-video_process.php“ a nalezneme jej v adresáři „html“. V prvé řadě skript ověří platnost vložených údajů – jestli nejsou položky názvu, popisu atd. prázdné a upraví se názvy pro případné hackerské útoky. Zároveň se také zkontroluje, jestli je nahraná soubor videem.

Asi nejčastější myšlenkou je omezení podle přípony souboru. Tedy dovolit pouze přípony *.avi, *.mpg atd. Problémem je však fakt, že mnoho dnešních uživatelů nemá o nějakých příponách vůbec tušení. Přispívá k tomu také operačním systém Windows, který v základním nastavení zcela skrývá přípony. A ačkoliv si to uživatel vůbec neuvědomuje, může se pokusit nahrát soubor „video.avi“, přičemž se vůbec nemusí jednat o soubor s videem.

Výše zmiňovaný systém detekce video-souboru tedy rozhodně není dokonalý a díky tomuto problému může nahrávací skript velice často zkolabovat. V tomto projektu je použito jiné – velice elegantní – řešení. Jedná se o detekci pomocí tzv. MIME typu. U každého nově nahraného souboru je zjištěn jeho MIME typ a pokud odpovídá videosouboru – tedy jeho MIME type je video/cokoliv – pak je soubor analyzován jako správný a skript pokračuje dál. V opačném případě je uživateli vrácena chybová hláška s informací, že nahraný soubor není video.

Po kontrole vstupních hodnot nahrávací skript pokračuje vložením informací do databáze – uloží se pouze nekompletní informace o videu, konkrétně se jedná o název, popis, ID uživatele, zvolá kategorie a klíčová slova.

V další fázi pak probíhá přesun nahraného souboru z dočasného úložiště skriptovacího jazyka do adresáře /data/upload/, kde je video pojmenováno podle ID v databázi.

V posledním kroku se spustí na pozadí převodní skript. Tento skript je nutné spouštět se specifickým nastavením PHP. Jak bylo mnohokrát zmíněno, konvertování trvá velice dlouho a při standardním nastavení php.ini může běžet jeden skript maximálně 30 sekund. Jelikož by se do tohoto limitu převod videa ve většině případů nevešel, je nutné nastavit hodnoty max_execution_time a max_input_time na nějaké vyšší hodnoty. Výsledný kód pro zavolání skriptu na pozadí může vypadat třeba takto (na OS Linux)

```
exec("php -d max_input_time=100000 -d max_execution_time=100000 -d
safe_mode=0 konvert.php $ID_souboru > /dev/null &");
```

7.5.2 Konvertovací skript

Podobně jako u skriptu na zpracování nahraného souboru i zde se na začátku ověřují vstupy. Mírnou odchylkou od klasických skriptů je zde zpracování proměnné. Jediná proměnná je ID souboru a není předávána metodou GET ani POST, nýbrž jako další argument při spouštění systému. Tato proměnná se následně zpracovává jako $argv[1].

Další částí je analyzování videa. Otevře se uživatelem nahraná soubor (v adresáři /data/upload/) a zjistí se informace o datovém toku videa. Tato informace je velice důležitá, neboť se na základě toho určuje, zda bude video konvertováno i do vysoké kvality (HD). Kritérium je pevně dané – pokud má zdrojové video datový tok 1.000.000 bit/s (cca 1 Mbit/s) a vyšší, pak je video označeno jako dostatečně kvalitní a bude se vytvářet i HD kvalita. V tomto kroku se také „vypočítá“ adresář, kam se bude video ukládat – tedy formát adresáře rok-týden, např. 2010-02.

Ihned po analyzování přichází na řadu konvertování videa. Prvně se vytvoří adresáře, kam budou data ukládána, v dalším kroku následuje samotné konvertování. Po dokončení konvertování do formátu FLV následuje úprava souboru pomocí programu flvtool2, který vloží do videa potřebná metadata (více o konvertování a úpravě FLV souboru bude napsáno dále). Jako úplně první příkaz v tomto „bloku“ je však odpojení od databáze. Není to úplně nutný krok, avšak je důrazně doporučen. Předchozí příkazy (konvertování) trvá desítky, někdy i stovky vteřin a po celou dobu by bylo aktivní spojení s databází, což není zrovna ideální. Takové připojení pak zbytečně vytěžuje systémové prostředky, v horším případě může vypršet platnost relace (připojení k databázi) a následné dotazy nebudou funkční a celý proces zkolabuje.

Dalším krokem v procesu konvertování videa je vytvoření náhledů. Tyto náhledy jsou použity v seznamu videí na webových stránkách a také jako náhled přímo v přehrávači videa (pokud není spuštěno přehrávání). Vytváří se dva náhledy. Jeden v originálním rozlišení videa, druhý – menší – s šířkou 200 px. Důvodem tohoto kroku je ušetření konektivity. Pokud by uživatel prohlížel náhledy v originálním rozlišení při zobrazení výpisu videí, bylo by to zbytečně zatěžující.

V poslední fázi se aktualizuje řádek v databázi, kam se vloží informace o tom, zda je video dostupné v HD kvalitě, vloží se délka (trvání) videa a hlavně se změní příznak videa „zkonvertovano“ na hodnotu 1, čímž je jasně dáno, že proces konvertování skončil a video je možné si prohlédnout. Všechny dotazy na stránce, které nějakým způsobem zobrazují video, pracují právě s tímto příznakem a vždy je testováno, jestli je video zkonvertováno. Pokud není, video se považuje za neexistující. Nakonec se odešle informativní email uživateli, že video bylo úspěšně převedeno a může jej zhlédnout.

7.5.3 Konvertování programem FFmpeg

FFmpeg je kolekce free programů sloužících pro nahrávání, konverzi a streamování audia a videa. Základem této kolekce je libavcodec, což je knihovna pro kompresi audia a videa. Oficiálně je FFmpeg vyvíjen v operačním systému Linux, je však možné jej zkompilovat ve většině operačních systému – vč. Windows. FFmpeg se skládá mj. z následujících komponent [11]

- ffmpeg – utilita pro příkazovou řádku umožňující konvertování videa a audia. Zároveň umožňuje grabování a enkódování v realném čase z televizní karty
- ffserver – streamovací server pro online vysílání, obsahuje mnoho chyb
- ffplay – jednoduchý multimediální přehrávač
- libavcodec – knihovna obsahující audio a video kodeky, tyto byly vyvinuty s důrazem na maximální výkon

Samotný FFmpeg – jakožto konvertovací program – má neskutečně mnoho parametrů a nastavení. Rozhodně není cílem této práce popisovat všechny dovednosti tohoto programu. Avšak ty nejdůležitější a nejzajímavější parametry si zde uvedeme [21]

- -h – zobrazí nápovědu
- -version – verze programu
- -formats – seznam dostupných formátů, příznak „D“ znamená, že kodek podporuje dekódování, příznak „E“ pak enkókodání
- -codecs – seznam dostupných kodeků
- -i – definice vstupního souboru (zdroj)
- -f – (násilné) definování formátu
- -y – v případě existence výstupního souboru jej bez dotázání přepíše
- -ss – „přetočení“ videa na danou pozici, konvertování tedy nemusí probíhat od začátku
- -t – jak dlouhý časový úsek se má konvertovat

- -dframes – počet snímků, které budou konvertovány
- -b – nastaví datový tok videa (v jednotkách bit/s), standardní hodnota je nastavena na 200 kbit/s
- -r – počet snímků za sekundu, standardně 25 fps.
- -s – nastavení rozlišení videa, je možné použít předdefinované konstanty, např. qvga (320x240), vga (640x480), qcif (176x144) atd. Standardně je rozlišení nastaveno stejné jako u zdrojového souboru.
- -aspect – poměr stran
- -croptop, -cropbottom, -cronleft, -cronright – nastaví oříznutí videa v pixelech
- -vn – video nebude zaznamenáno
- -vcodec – definování použitého videokodeku pro převod
- -ar – nastavení frekvence audia (44010 Hz, 22050 Hz atd.)
- -ab – datový tok zvuku (v jednotkách kbit/s), standardně 64 kbit/s
- -ac – počet zvukových kanálu (2=stereo atd.), výchozí hodnota = 1
- -an – zvuk nebude zaznamenán
- -acodec – nastavení audiokodeku
- -benchmark – po skončení konvertování zobrazí informace o zátěži při samotném procesu, např. vytížení procesoru, nejvíce využívané paměti atd.
- -vpre – definuje použití tzv. profilů kvality, bližší popis byl vysvětlen v subkapitole výběr kodeku, konkrétně H.264

Na první pohled se může zdát ovládání programu FFmpeg v příkazové řadce (shellu) velice obtížné, avšak opak je pravdou. Po zažití syntaxe je používání nesmírně rychlé, obrovskou výhodou je právě ovládání přes příkazy, umožňuje vytvoření jednoduchých skriptů pro hromadnou konverzi atd.

Pro úplnost několik příkladů s vysvětlením

- `ffmpeg -codecs`
  vypsání všech dostupných kodeků
- `ffmpeg -i video.avi zvuk.mp3`
  elementární extrahování zvuku z videosouboru – FFmpeg je inteligentní a na základě přípony výstupního souboru (mp3) určí správný kodek – např. libmp3lame

- `ffmpeg -i video.avi –acodec aac -ac 2 -ab 128k -ar 44100 zvuk.aac`
  pokročilejší extrahování zvuku, nastaven kodek AAC, 2 zvukové kanály a datový tok 128 kbit/s

- `ffmpeg -i video.avi -b 400k -ab 64k -s qvga video2.avi`
  zdrojový soubor převede do stejného formátu, avšak s datovým tokem videa 400 kbit/s, zvukovým datovým tokem pak 64 kbit/s, výsledný datový tok by měl tedy být cca 464 kbit/s. Zároveň bude rozlišení videa převedeno na 320x240 bodů

- `ffmpeg -i video.avi -vframes 1 –ss 0:0:30 –f image2 nahled.png`
  velice jednoduchý způsob, jak vytvořit z videa náhled. Parametr „vframes“ určuje zpracování pouze jednoho videosnímku a parametr „ss“ „přetočí“ video do místa 30 sekund

V praktické práci jsou používány dvě konvertování pomocí programu FFmpeg, jednou pro převod videa do formátu FLV. Zvolen byl datový tok 500 kbit/s pro obraz, pro zvuk 64 kbit/s, výsledný příkaz vypadá následovně

```
ffmpeg -i $vstupni_soubor -b 500k -vcodec flv -ab 64k -ar 22050 -acodec
mp3 $vystupni_soubor.flv
```

Pokud je zdrojové video dostatečně kvalitní, je vytvářen také kvalitní HD výstup. Pro tento případ byl zvolen kodek H.264 s bitrate 2000 kbit/s, zvuk je kódován pomocí kvalitního kodeku AAC. Jako profil kodeku H.264 byl zvolen „default“, jenž dosáhl v testování této práce výborného poměru kvalita/rychlost.

```
ffmpeg -i $vstupni_soubor -b 2000k -vcodec libx264 -vpre default -ab 128k
-acodec aac $vystupni_soubor.flv
```

7.5.4 Úprava FLV videí

Převod programem FFmpeg má u formátu FLV jeden zásadní nedostatek – součástí výsledného videa nejsou metadata. Není to až tak zásadní problém, avšak pokud bychom chtěli využívat tzv. funkci seekování, nebude fungovat. Tato funkce umožňuje „přetáčet“ video ve Flash přehrávači na libovolné místo a to bez nutnosti video celé načíst. Přidá se tím uživatelský komfort – pokud návštěvník nedisponuje rychlou internetovou linkou a chce si dlouhé video prohlížet až od 10. minuty, pak to nejde. V neposlední řadě se díky seekování ušetří nemalá část přenosu dat.

Pokud chceme metadata do FLV videa vložit, je nutné použít nějaký specializovaný program. V tomto projektu je využíváno programu flvtool2. Program umožňuje mnohem víc operací s FLV videem, pro naše potřeby však dostačuje zcela elementární příkaz, který video zanalyzuje a přidá do souboru tolik tížená metadata.

```
flvtool2 -U video.flv
```

7.5.5 Generování náhledů videa

V příkladech programu FFmpeg o pár řádků výš byl uveden příkaz, pomocí jehož lze velice snadno vytvořit náhled z videa. Vytváření touto metodou je však relativně dost náročné na zátěž serveru. Avšak není problém tento způsob využívat.

V této práci je použito jiné řešení. Konkrétně je použito rozšíření ffmpeg-php, které umožňuje vytváření náhledů také. A nejen to, jedná se knihovnu funkcí pro skriptovací jazyk PHP, kterou si lze velice snadno ulehčit práci. V kódu projektu je tato knihovna několikrát použita, konkrétně např. pro zjištění datového toku videa, na základě kterého se určuje, zdali se bude vytvářet HD kvalita nebo pro zjištění délky (trvání) videa. Samotný kód pro vytváření náhledu, vč. jeho popisu, je uveden v příloze D.

7.6 Streamování videa

Videa jako taková jsou běžným způsobem čtena pomocí Flash přehrávače ze serveru a následně posílána uživateli (přehrávači), aby si mohl obsah přehrát. To je řešení, které funguje a není nijak náročné – stačí jakýkoliv webserver. Avšak má to svá úskalí. V části, kde byla rozebírána úprava FLV videí bylo zmíněno tzv. seekování. A problémem je, že na běžném webserveru toto seekování nefunguje – třebaže video obsahuje tolik potřebná metadata.

Klasickým způsobem je video nahráváno jako jeden celý soubor – od začátku do konce. Ovšem problém nastává, když chceme soubor stahovat třeba od 50 % (tam, kde si klikneme v přehrávači). To bohužel webservery neumějí.

Řešení je však relativně prosté – doinstalovat do webserveru speciální modul, který tuto „chybu“ vyřeší. Pro apache jsou to konkrétně moduly „mod_flvx“ pro podporu streamování FLV formátu a „h264_streaming_module“ pro podporu H.264 kodeku. Pro webový server lighttpd jsou to moduly „mod_flv_streaming“, resp. „mod_h264_streaming“. Pro úplnost ještě třetí webserver – nginx, zde se streamovací moduly jmenují „ngx_http_flv_module“ a „mod_mp4_streaming_lite“.

7.6.1 Shrnutí

Pokud chceme, aby náš videoserver umožňoval funkci tzv. seekování, je nutné

- mít na webserveru doinstalovanou podporu streamování FLV, resp. H.264
- mít FLV soubory opatřené metadaty – např. pomocí programu flvtool2 (u MP4 formátu s kodekem H.264 není nutné videa upravovat).

7.7 Návrhy na vylepšení

Ne nadarmo platí pravidlo, že fantazii se meze nekladou. Ani v tomto případě tomu není jinak. Vymyslet se dá prakticky cokoliv. Inspirací budiž současné videoservery, které se doslova předhánějí v tom, který uvede lepší a „vychytanější“ funkce.

Pokud pomineme funkce, které nemají s videoserverem jako takovým prakticky nic společného – chatování mezi uživateli, posílání soukromých zpráv atd., je zde několik návrhů, které přímo souvisí s videoserverem jako takovým. Se všemi zde uvedenými návrhy mám zkušenost a osobně jsem je implementoval na některý ze svých předchozích projektů.

7.7.1 Ochrana videosouborů

Každý autor, ať se jedná o autora čehokoliv, nemá rád, když jeho práci někdo vydává za svoji – zkrátka a jasně, práce je ukradena. I v případě videoserverů se takové případy stávají a existuje mnoho stránek, které „parazitují“ na něčích stránkách. Ačkoliv je video přehráváno na cizích stránkách, soubor jako takový je stahován z našeho serveru. Tím dochází ke dvěma nepříjemným vlastnostem – zvyšuje se přenos dat na serveru a video je „ukradeno“. Náš web tak přichází o návštěvníky, což mohou být zisky z reklamy apod.

Řešením je zabezpečit videa, resp. jejich odkazy. Normální odkaz ve tvaru www.neco.cz/videa/dovolena.flv jasně říká, kde se video na serveru nachází a je možné na něj libovolně odkazovat. Existují však speciální rozšíření webserverů, které generují odkazy pomocí hashe a výsledný odkaz je pak nesrozumitelný a absolutně nic neříká fyzickém umístění na serveru. Takto chráněný odkaz (ekvivalent k výše uvedenému odkazu) může vypadat třeba takto: www.neco.cz/ab18c6f1e061c5a6d1453e50f49984bb/af4fb4/dovolena.avi.

Nebudeme se zde zabývat konkrétním popisem, jak to přesně funguje, to je dostupné v příslušných manuálech. Tyto moduly se jmenují – pro apache je to „mod_auth_token“, v lighttpd „mod_secdownload“ a v nginx „http_secure_link_module“.

7.7.2 Generování více náhledů

Velice užitečnou funkcí může být vytváření více náhledů z videa. Pokud by se u každého videa vygenerovalo např. 10 náhledů, po najetí ukazatele myši na náhled např. na hlavní stránce, by se začaly tyto náhledy střídat. Návštěvník by tak měl mnohem větší přehled o tom, co se v daném videi nachází. A to bez nutnosti si video přehrát.

Jiným využitím může být výběr hlavního náhledu. Ne vždy je náhled videa povedený – např. přechod scény a jenom černo. V případě, že by bylo k dispozici více náhledů, by si mohl uživatel, který video nahrál, vybrat jeden z mnoha náhledů, který se bude používat.

7.7.3 Zpoplatnění obsahu

Po přečtení této práce je asi každému jasné, že provozování videoserveru není asi nejlevnější záležitost. Proto se jeví jako velice dobré řešení zpoplatňovat obsah a vydělat si tím na provoz serveru atd. Možností platby je spousta – ať už běžný převod z účtu na účet, metoda PayPal, Paysec nebo dnes velice rozvíjející se platební systémem – SMS zprávy. Ať si majitel serveru vybere cokoliv (nebo i klidně všechny varianty), může zavést do

systému např. nějaký kreditní systém, kdy každé zhlédnutí videa odečte 1 kredit atd. Možností je zde opět velice mnoho.

8 Porovnání s jinými videoservery

V praktické části této práce byl vytvořen videoserver, který bude v této kapitole srovnán s dostupnými projekty na internetu. Srovnání bude na dvou úrovních – uživatelské a technické. Největším kandidátem na srovnání je samozřejmě největší videoserver – youtube.

8.1 Uživatelské srovnání

Po uživatelské stránce jsou koncepty obou stránek zcela totožné. Videa nahrávají sami uživatelé, jedná se tedy o web 2.0, kde obsah tvoří návštěvníci. V základním modelu jsou oba dva videoservery opět zcela totožné – přehrávat videa může každý, nahrávat mohou pouze registrovaní, stejně tak komentovat videa. Tím však možnosti projektu v rámci této práce končí. Rozhodně nebylo úkolem ani záměrem vytvořit megalomanský projekt – kterým youtube bezesporu je. Youtube nabízí neskutečné množství funkcí navíc. Ať je to třeba sledován uživatelských kanálů, vytváření playlistů a nebo třeba překladem stránek do desítek světových jazyků.

Komfortem však projekt v této práci nijak nezaostává. Používá moderní technologie (ajax), které umožňují nerušené sledování videa a přitom bezproblémové hlasování a přidávání komentářů bez nutnosti obnovení stránky. Vyhledávání je také velice schopné a propracované. Avšak vše uvedené je u youtube jakási samozřejmost.

Hlavním lákadlem, kterému nemůže konkurovat žádná stránka na světě, je samozřejmě obsah. Youtube nabízí miliony videí a proto je tak oblíbený.

8.2 Technické srovnání

„Pozadí“ videoserveru, tedy to, co běžný uživatel nevidí, resp. si neuvědomuje, je na velice dobré úrovni. Už od počátku vývoje byl kladen důraz na kvalitu a především výkonnost projektu. Je trochu těžké porovnávat ono „pozadí“, neboť do samotného jádra vidí jen a pouze vývojáři daného projektu.

Asi nejprimárnějším aspektem je kvalita videa. Zde se dá říct, že youtube je na stejné kvalitě jako tento videoserver, neboť jsou používány stejné kodeky (H.264). youtube pak nabízí více formátů videa – 360p, 480p, 720p a 1080p. V případě tohoto videoserveru je k dispozici pouze formát FLV s 500kbit/s datovým tokem, popř. HD kvalita – kodek H.264 s bitrate 2000 kbit/s.

Funkce přehrávače jsou velice totožné, přehrávač v tomto videoserveru do jisté míry (po použití volitelných pluginů) ten na youtube překonává. Je to však dáno zvoleným – vysoce kvalitním – přehrávačem JW Flash Player.

Jelikož je na server, kde běží tento projekt, nainstalováno vše potřebné – rozšíření pro streamování videa i úprava videí programem flvtool2, funguje i tzv. seekování, tedy

„přetáčení“ videa do libovolné pozice bez nutnosti mít načtený celý soubor. Na youtube je toto opět samozřejmostí.

Youtube pak obsahuje mnoho dalších vymožeností – podobně jako u uživatelského srovnání a projekt této práce mu nemá šanci konkurovat. Namátkou je to třeba podpora autorských práv, tzv. DRM. Každé nově nahrané video je kontrolováno pomocí této technologie a pokud je nalezena shoda – tedy zvuk je licencován (obvykle písničky) – je u videa potlačen zvuk. Youtube má také vlastní API rozhraní, díky kterému nejednomu vývojáři ulehčí život. Obrovskou „vychytávkou“ je opravování slov při hledání. Pokud hledáme slovo „auto“ a uděláme překlep a místo toho napíšeme třeba „atuo“, youtube hned zobrazí, jestli jsme se neupsali.

I v této části si na konec necháme obsahovou část. Díky mnoha miliónům videí musí mít youtube zcela specifické hardwarové vybavení, odhadem to musí být stovky, možná tisíce serverů, které obstarávají chod této jedné stránky. Tomu se nemůže rovnat žádný jiný videoserver. Na druhou stranu by nebylo až tak složité rozšířit zde publikovaný projekt, aby také zvládal pracovat na více serverech.

8.3 Shrnutí

Youtube je samozřejmě číslo jedna v serverech a je to vzor pro všechny ostatní. V porovnání s ostatními – menšími – videoserver je tento projekt velice schopným konkurentem. Ačkoliv obsahuje pouze základní model fungování videoserveru a žádné doplňkové funkce navíc, jeho jádro je tvořeno s velikou rozvahou a s velikým potenciálem.

Mnoho dnešních – dokonce i dost úspěšných – videoserverů nemá možnost seekování nebo nepoužívá moderní technologii ajax pro odesílání hodnocení, komentářů atd.

9 Závěr

Při zpracování teoretické části této práce jsem nabyl mnoha nových informací, rozšířil jsem si znalosti na poli videoserverů, především přínosný pro mě bylo srovnávání jednotlivých formátů a kodeků. Bylo velice zajímavé zkoumat, jak je výsledné video ovlivňováno mnohými parametry a jak se rapidně mění časy potřebné pro převod videa. V neposlední řadě jsem byl velice rád, že jsem mohl v této práci uplatnit svoje předchozí znalosti v oblasti videoserverů, především pak v části, kde byla popisována zátěž v závislosti na počtu návštěvníků. Pevně věřím, že čtenářům, jež si tuto práci přečetli, byla hodnotným přínosem a nabyli nových informací.

Výsledná praktická práce byla vytvářena od samotného začátku jako velice promyšlený projekt řešící i ty nejmenší detaily. Rozhodně se nejedná o nějaký megalomanský projekt, který obsahuje úplně všechno, na co si jenom vzpomeneme. Jedná se o základní videoserver, který je však postaven na kvalitním jádře a je připraven do ostrého provozu. Jsou zde vyřešeny otázky bezpečnosti, komfortu uživatele a především toho hlavního – funkčnosti. Web je plně funkční a ihned připraven k provozu.

Literatura

[1] *Digital Ethnography @ KSU* [online]. 2008 [cit. 2010-05-09]. YouTube Statistics. Dostupné z WWW: <http://ksudigg.wetpaint.com/page/YouTube+Statistics>.

[2] Adobe Flash In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 10 January 2002, 7 May 2010 [cit. 2010-05-09]. Dostupné z WWW: <http://en.wikipedia.org/wiki/Adobe_Flash>.

[3] *Adobe.com* [online]. 2010 [cit. 2010-05-09]. Flash Player Statistics. Dostupné z WWW: <http://www.adobe.com/products/player_census/flashplayer/>.

[4] Youtube In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 8 February 2006, 29 April 2009 [cit. 2010-05-09]. Dostupné z WWW: <http://en.wikipedia.org/wiki/Youtube>.

[5] *LUPA* [online]. 2008 [cit. 2010-05-09]. Pořady Televize Prima na Streamu - premiéry na webu již druhý den. Dostupné z WWW: <http://www.lupa.cz/tiskove-zpravy/porady-televize-prima-na-streamu/>.

[6] Silverlight In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 16 April 2007, 25 January 2009 [cit. 2010-05-09]. Dostupné z WWW: <http://en.wikipedia.org/wiki/Silverlight>.

[7] HTML5 video In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 7 February 2010, 7 May 2010 [cit. 2010-05-09]. Dostupné z WWW: <http://en.wikipedia.org/wiki/HTML5_video>.

[8] Kodek In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 9. 12. 2004, 4. 5. 2010 [cit. 2010-05-09]. Dostupné z WWW: <http://cs.wikipedia.org/wiki/Kodek>.

[9] Codec In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 12 November 2001 , 4 May 2010 [cit. 2010-05-09]. Dostupné z WWW: <http://en.wikipedia.org/wiki/Codec>.

[10] Multimediální kontejner In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 20. 5. 2008, 16. 3. 2010 [cit. 2010-05-09]. Dostupné z WWW: <http://cs.wikipedia.org/wiki/Multimedi%C3%A1ln%C3%AD_kontejner>.

[11] FFmpeg In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 22 April 2003, 27 April 2010 [cit. 2010-05-09]. Dostupné z WWW: <http://en.wikipedia.org/wiki/FFmpeg>.

[12] FLV In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 2 May 2007, 2 May 2007 [cit. 2010-05-09]. Dostupné z WWW: <http://en.wikipedia.org/wiki/FLV>.

[13] *Open Source Flash* [online]. 2008 [cit. 2010-05-09]. Flash Video (FLV). Dostupné z WWW: <http://osflash.org/flv>.

[14] VP6 In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 1 December 2003, 11 July 2009 [cit. 2010-05-09]. Dostupné z WWW: <http://en.wikipedia.org/wiki/VP6>.

[15] MPEG-4 Part 14 In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 10 March 2004, 29 April 2010 [cit. 2010-05-09]. Dostupné z WWW: <http://en.wikipedia.org/wiki/MPEG-4_Part_14>.

[16] H.264 In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 1 April 2005, 2 November 2005 [cit. 2010-05-09]. Dostupné z WWW: <http://en.wikipedia.org/wiki/H.264/MPEG-4_AVC>.

[17] Mp3 In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 27 July 2001, 8 December 2007 [cit. 2010-05-09]. Dostupné z WWW: <http://en.wikipedia.org/wiki/Mp3>.

[18] Advanced Audio Coding In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 29 April 2003, 3 May 2010 [cit. 2010-05-09]. Dostupné z WWW: <http://en.wikipedia.org/wiki/Advanced_Audio_Coding>.

[19] *LongTail Video* [online]. 2010 [cit. 2010-05-09]. Home of the JW Player. Dostupné z WWW: <http://www.longtailvideo.com/>.

[20] *Longtail* [online]. 2010 [cit. 2010-05-09]. AddOns | Plugins. Dostupné z WWW: <http://www.longtailvideo.com/addons/plugins>.

[21] *Ffmpeg.org* [online]. 2010 [cit. 2010-05-09]. FFmpeg Documentation. Dostupné z WWW: <http://ffmpeg.org/ffmpeg-doc.html>.

Příloha A – Elementární kód pro vložení JW Playeru na stránku

Níže uvedený kód počítá s přítomností souborů player.swf (přehrávač), swfobject.js (knihovna pro vkládání swf aplikací) a video.flv (video, které bude přehráno) ve stejném adresáři, kde se bude nacházet výsledný html soubor

```
<script type='text/javascript' src='swfobject.js'></script>
<div id='mediaspace'>Zde bude přehrávač</div>
<script type='text/javascript'>
var so = new SWFObject('player.swf','ply','470','320','9','#ffffff');
so.addParam('allowfullscreen','true');
so.addParam('allowscriptaccess','always');
so.addParam('wmode','opaque');
so.addVariable('file',‘video.flv');
so.write('mediaspace');
</script>
```

Příloha B – SQL dotaz používaný pro vyhledávání

```
SELECT
      V.ID,
      V.nazev,
      V.datum,
      V.HD,
      DATE_FORMAT(V.datum, '%d.%m.%Y %H:%i') AS datum_format,
      V.popis,
      DATE_FORMAT(V.datum, '%Y-%u') AS adresar,
      IFNULL(ROUND(V.hodnoceni_soucet/V.hodnoceni_pocet), 0) AS
hodnoceni,
      V.zobrazeni,
      V.ID_uzivatele AS U_ID,
      U.jmeno AS U_jmeno,
      V.ID_kategorie AS K_ID,
      K.nazev AS K_nazev,
      (SELECT COUNT(*) FROM komentare WHERE ID_videa=V.ID) AS
pocet_komentaru
FROM videa V
LEFT JOIN kategorie K ON K.ID=V.ID_kategorie
LEFT JOIN uzivatele U ON U.ID=V.ID_uzivatele
WHERE
      zkonvertovano=1 AND
      (MATCH (V.nazev) AGAINST ('$text_strip' IN BOOLEAN MODE) OR
      MATCH (V.popis) AGAINST ('$text_strip' IN BOOLEAN MODE) OR
V.ID IN (SELECT ID_videa FROM klicova_slova WHERE MATCH(slovo) AGAINST
('$text_strip' IN BOOLEAN MODE)))
```

Příloha C – Adresářová struktura

Struktura adresářů je volena s důrazem na co největší přehlednost. A asi nikoho nepřekvapí, že ve složce /js/ najdeme JavaScriptové funkce, ve složce /obrazky/ bude grafika pro web a v adresáři /html/ budou php soubory pro chod aplikace.

Co je však podstatnější, to je adresář /data/, kam se ukládají videa a také náhledy videí. V tomto adresáři najdeme další čtyři

- nahledy
- upload
- video-flv
- video-mp4

Obsah je zajisté opět zřejmý, avšak videa nejsou v adresářích /video-flv/ a /video-mp4/ umístěna na „jedné hromadě“. Videa jsou dále tříděna ve složkách podle roku a týdne nahrání. Pokud bylo tedy video nahráno 10.1.2010, pak bude v adresáři /video-flv/2010-02/ (druhý týden). Důvod tohoto řešení je prostý. V případě, že by na webu bylo obrovské množství videí (řádově tisíce až desetitisíce), operačnímu systému by mohlo trvat poměrně hodně času, než by obsah adresáře načetl. Přidání těchto subadresářů má tedy také jakýsi výkonnostní účel.

Příloha D – kód pro generování náhledů z videa

```
$mov = new ffmpeg_movie("video.flv", false);
$pocet_frames = $mov->getFrameCount();
$frame = $mov->getFrame( round($pocet_frames * 0.5) );
$gd = $frame->toGDImage();
imagejpeg($gd, "nahled.jpg");
imagedestroy($gd);
```

Bez předchozí znalosti této knihovny je nejspíše všem zřejmé, co který řádek dělá. V prvním kroku se načte video, následně se celkový zjistí počet snímků, nastaví se aktuální snímek na 50 % délky videa – tedy počet snímků * 0,5. Ve výsledku se vybraný snímek převede na klasický obrázek grafické knihovny jazyka PHP a uloží se jako jpg.