UNIVERZITA PARDUBICE
ÚSTAV ELEKTROTECHNIKY A INFORMATIKY

ELEKTRONICKÝ PRODEJNÍ SYSTÉM
BAKALÁŘSKÁ PRÁCE

AUTOR PRÁCE: Luboš Horák
VEDOUCÍ PRÁCE: Ing. Lukáš Čegan
2007

UNIVERSITY OF PARDUBICE
INSTITUTE OF ELECTRICAL ENGINEERING AND INFORMATICS

ELEKTRONICALLY SELLING SYSTEM
BACHELOR WORK

AUTHOR: Luboš Horák
SUPERVISOR: Ing. Lukáš Čegan

2007

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Elektronický prodejní systém

STUDIJNÍ OBOR
Informační technologie

VEDOUCÍ
Ing. Lukáš Čegan
Ústav elektrotechniky a informatiky
Univerzita Pardubice

Zásady pro zpracování

Teoretická část bakalářské práce se bude zabývat analýzou současných technologických možností elektronických prodejních systémů. Budou zhodnoceny možnosti nasazení komerčních řešení dostupných na našem trhu v porovnání s možností vývoje vlastního řešení.

V implementační části bakalářské práce bude provedena analýza současného prodejního systému konkrétní společnosti na základě které budou navrhnuty a následně naprogramovány nové moduly, které zváží celkovou funkcionalitu systému a které budou odrážet současné vývojové trendy elektronického obchodování.

Prohlašuji:

Tuto práci jsem 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 _. 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 Univerzity Pardubice.

V Pardubicích dne 12. 5. 2007

Luboš Horák

ABSTRAKT

V této práci se zabývám problematikou internetového obchodování a moderních trendů v tomto oboru. První část věnuji teoretickému základu, kde popíšu základní technologie, které se používají.

V praktické části řeším integraci na úrovni dat. V mém případě je to nahrávání ceníků od tří nezávislých společností, z nichž každá z nich poskytuje svůj ceník v jiném formátu (databáze Microsoft Access, CSV, XML). Dále řeším problematiku velké četnosti vazeb.

ÚDAJE PRO KNIHOVNICKOU DATABÁZI

Název práce	Elektronický prodejní systém
Autor práce	Luboš Horák
Obor	Informační technologie
Rok obhajoby	2007
Vedoucí práce	Ing. Lukáš Čegan
Anotace	
Klíčová slova	Internetové obchodování, Integrace dat, Bezpečnost na internetu

1. Úvod

V této práci se budu zabývat elektronickým obchodováním a moderními trendy v tomto oboru informační technologie. V první fázi představím teoretický základ, kde se hlavně zaměřím na historii elektronického obchodování a vysvětlení pojmu e-Commerce. Dále pak popíšu architekturu webové databázové aplikace, pomocí níž se dnes řeší e-shopy a její jednotlivé části, což znamená, že vylíčím funkce a vlastnosti nejvyužívanějších webových serverů, programovacích jazyků a uvedu zde několik způsobů, jak dělíme databázové systémy. Na základě těchto kriterií pak porovnáme mezi sebou 3 nejoblíbenější databázové systémy. Následně okrajově nastíním základní technologie v oblasti bezpečnosti přenosu dat, což je jedna z nejdůležitějších oblastí v problematice internetového obchodování. Porovnám mezi sebou i dostupná řešení e-shopů.

Ve druhé fázi rozeberu současný stav internetového obchodu společnosti B2Com, kde naznačím špatný návrh integrování ceníků do databáze, na němž uvidíme jednoznačnou nerozšiřitelnost. Poté nastíním návrh nového řešení, který odstraní ty nejdůležitější chyby starého schématu a který by ho měl udělat o mnoho flexibilnějším. Rozeberu zde i několik metod integrace dat, přičemž zhodnotím, která metoda bude pro tento projekt nejefektivnější.

V poslední fázi se zaměřím na samotnou implementaci – tzn. návrh databáze, řešení duplikátů samotných produktů, při nahrávání ceníků od jednotlivých výrobců, kteří poskytují své ceníky v různých formátech. Zde popíšu postup nahrávání z formátů – CSV („Comma Separated Values“), MDB (databáze Microsoft Access) a nakonec nahrávání pomocí webových služeb, které nám na dotazy poskytují soubory XML („*eXtensible Markup Language*“). Tyto tři formáty jsou nejčastěji používanými v současných trendech internetového obchodování.

2. Elektronické obchodování

2. 1. Historie elektronického obchodování

Mezi prvními průkopníky elektronického obchodování musíme zařadit systém EDI (Elektronic Data Interchange), jež se začal užívat již v průběhu 80. let. Tento systém byl ovšem nasazován jen u velkých podniků, které realizovaly obrovská množství transakcí. Typickým příkladem je automobilový průmysl. Uvažme, jak komplikované bylo zpracování každé jednotlivé dodávky. Odběratel vytiskl objednávku, zaslal ji poštou nebo faxem dodavateli. Ten ji musel manuálně přepsat do počítačové evidence.

Je jasné, že tento způsob je nejen pomalý, ale hlavně drahý. Nepočítáme-li poplatky za poštovné či telefon, poplatky za potištěný papír atd., největším nákladem byla určitě lidská práce potřebná na zpracování odeslaných a došlých dokumentů.

Není tedy divu, že velké podniky začali využívat informační systémy EDI, které tyto náklady dramaticky snížily. Odběratel zapsal objednávku do svého informačního systému, ta byla v zápětí odeslána v elektronické podobě a bez nutnosti lidského zásahu zaevidována do systému dodavatele. Tato myšlenka byla jistým předchůdcem dnešního extranetu a od něj je už jen malý krůček k dnešnímu pojetí elektronického obchodování, a to k e-Commerce.

2. 2. e-Commerce

Tímto termínem (také e-komerce, e-obchod) se obvykle označuje marketingová a obchodní komunikace se zákazníky a vlastní obchodování, využívající jako hlavní nástroj Internet, a snaha pomocí něho využít nejmodernější informační a komunikační technologie ke zvýšení efektivity vztahu mezi podnikem a jeho zákazníky navzájem. Zásadním sledovaným účelem je samozřejmě zvýšení obratu a zisku rozšířením klientské základny, úspora nákladů, zlepšení image společnosti. Na rozdíl od všeobecně známé e-mailové komunikace, e-commerce zahrnuje nejen elektronický přenos informací a dokumentů,

ale především samotné uzavírání obchodních smluv prostřednictvím Internetu.

3. Architektura webové databázové aplikace

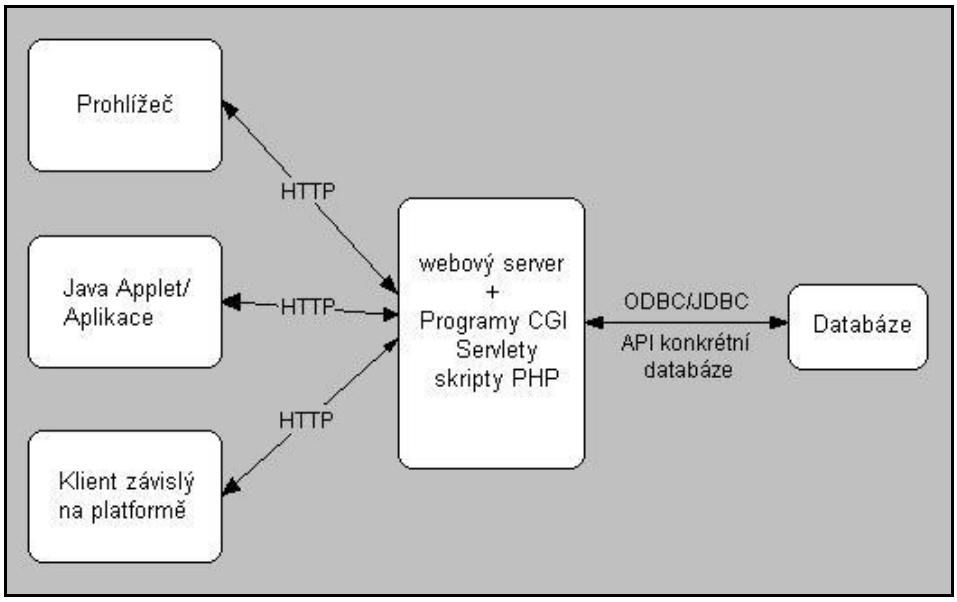
Základními vrstvami databázové webové aplikace jsou (přibližný náčrt vidíme na obrázku č. 1):

⇒ klient: uživatelův webový prohlížeč, java applet, aplikace java;

⇒ logika aplikace: zakódovaná do algoritmů požívaných ve skriptech CGI, speciálních modulech webového serveru nebo dokonce v serveru závislém na aplikaci;

⇒ databázová konektivita:API databáze nebo obecné propojovací protokoly, jako např. ODBC nebo JDBC;

⇒ databázový server: RDBMS, ODBMS.



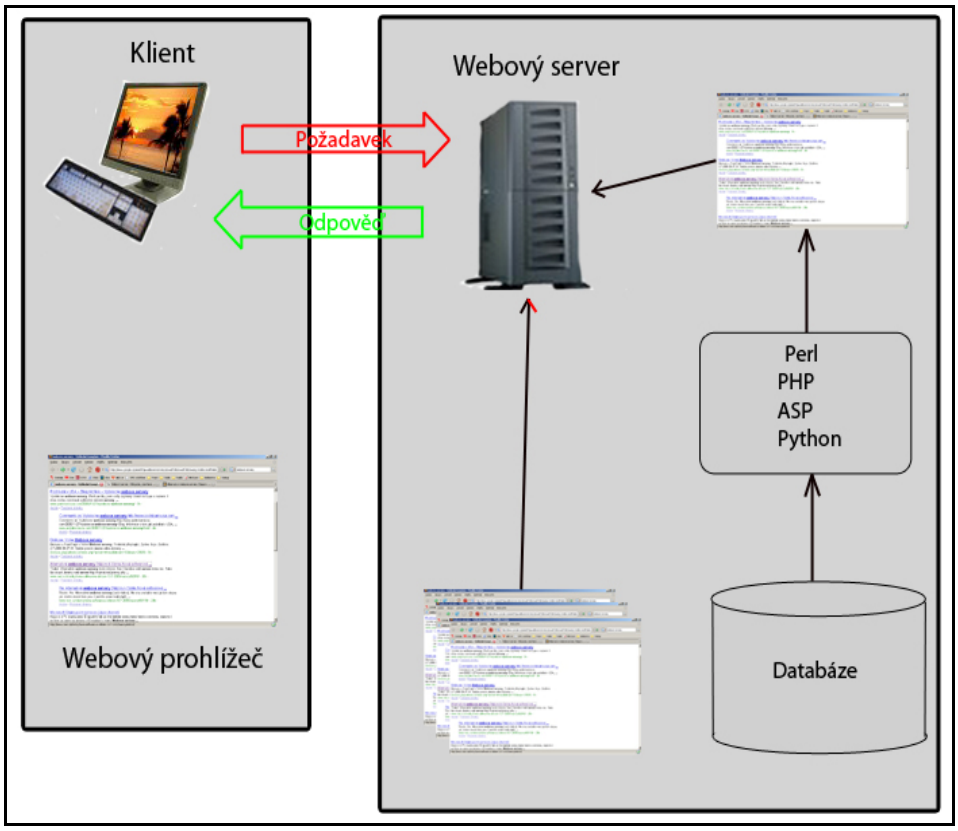
Obrázek č. 1 – Architektura webové aplikace (1)

4. Webové servery

4. 1. Funkce webového serveru

a. Počítač, který vyřizuje požadavky HTTP od klientů. Za vyřízení požadavku se považuje odeslaní webové stránky, což je názorně vidět na obrázku č. 2;

b. Počítačový program, jenž zpracovává požadavky z předchozího bodu.



Obrázek č. 2 – Princip webového serveru

4. 2. *Příklady serverů*

4. 2. 1. Apache HTTP Server

je softwarový webový server z kategorie open source pro Linux, BSD, Microsoft Windows a další platformy. Dá se říct, že Apache ve své kategorii nemá díky svým vlastnostem konkurenci.

Původ Apache se traduje na rok 1993, kdy se začal vyvíjet v NCSA (National Center for Supercomputing Aplications) na Illinoiské univerzitě. Původně se jmenoval NCSA HTTPd. V následujícím roce však opustil vývojářský tým hlavní programátor Rob McCool a tím došlo ke zpomalení vývoje. V roce 1998 se jeho vývoj úplně zastavil. Ovšem mezitím již NCSA HTTPd používali správci webových serverů a ti k němu dodávali vlastni úpravy. Nejdůležitější z nich se ukázali Brian Behlendorf a Cliff Skolnick, kteří založili e-mailovou konferenci a začali koordinovat sběr jednotlivých úprav.

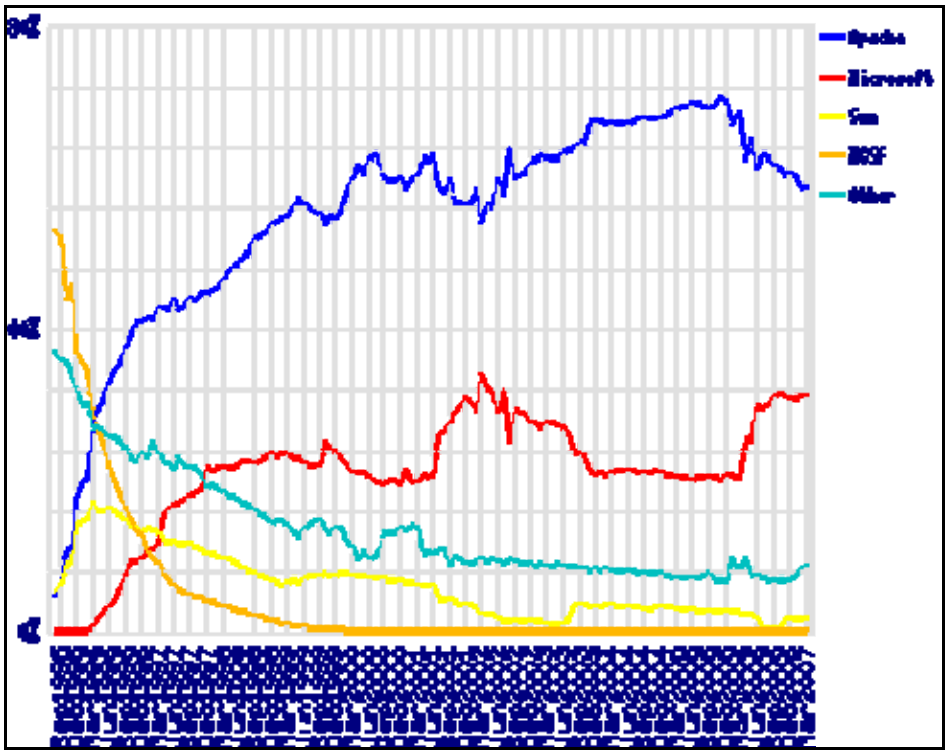
V roce 1995 se vydala první veřejná verze a nesla označení 0.6.2. Poté následovalo úplné přepsání kódu => Apache2 již neobsahuje

naprosto nic z původního NSCA HTTPd. Také byl založen Apache Group, který je dnes hlavní částí vývojářského týmu.

4. 2. 2. MS IIS server

Internet Information Server je čím dál populárnější server z dílny Microsoftu. Tento zajímavý web server je vysoce spolehlivý a snadno ovladatelný. Bohužel není multiplatformní – běží pouze na systémech Windows. MS Windows Server 2003 již v základní instalaci obsahuje IIS verze 6.0.

Na obrázku č. 3 je graf, na němž je znázorněn poměr využívání těch nejdůležitějších webových serverů od roku 1995 až po současnost. Modrá čára představuje námi prvně zmiňovaný Apache a červená MS IIS, z čehož vidíme jak tyto dva jasně dominují.



Obrázek č. 3 – Využívání webových serverů (2)

5. Programovací jazyky

Další důležitou fází je výběr programovacího jazyka, kterým můžeme projekt realizovat. Jazyky mohou být interpretované (kód se obecně převádí do bajtového kódu, který pak interpret jazyka převádí na instrukce procesoru) nebo kompilované (kód se jednou zkompiluje do

instrukcí procesoru a ten je pak vykonává), dále strukturované nebo objektově orientované atd...

5. 1. 1. ASP.NET

Je to pokračování programovacího jazyka ASP, ale pracuje to naprosto rozdílně. ASP vykonávalo bloky od shora dolů a kód <% %> ve výkonných blocích byl posílán na výstup. ASP.NET stránky jsou kompilovány do IL kódu a poté do nativního kódu Just-in-time kompilátory. Pak je kód vykonáván plnou rychlostí.

Tato technologie funguje nad .NET Framework, který poskytuje systémové služby, které podporuje ASP.NET a též přidává do systému základní podpůrné služby pro .NET technologie (vše je volně stažitelné na stránkách Microsoftu – asi 20 MB).

5. 1. 2. Perl

Perl – Practical Extraction and Report Language. Autorem je Kanaďan Larry Wall. První verze se pro veřejnost uvolnila v roce 1987 ve verzi 1.0.

Je to výborný skriptovací jazyk s perfektní podporou práce s řetězci. Programátorům, jenž umí v Javě, se v něm programuje rozhodně lépe než v shellu a navíc se v něm dají velice snadno i externí utility.

Perl se zařazuje do vyšších programovacích jazyků (má i podporu OOP), ale přesto si uchovává blízkost systému, protože vychází z UNIXovských shellů a jazyka C, tj. z nástrojů, na které jsou administrátoři zvyklí. Spadá pod GNU GPL Actistic licenci => může se používat zdarma a to i v komerčních projektech. Používá se jako vynikající nástroj pro CGI skriptování (CGI je externí program, který je spouštěn na žádost WWW serveru). Je multiplatformní a má skvělé nástroje pro práci s textem, pro práci s databázemi a pro síťové prostředí. Je vhodný pro nejrůznější úpravy na operačních systémech Windows, kde nahrazuje chybějící skriptovací administrátorský jazyk (můžeme za něj považovat Visual Basic, který ovšem není na ostatních platformách k dispozici). Samozřejmě má i několik nevýhod, mezi které někteří programátoři počítají, že se nemusí deklarovat proměnné, není zde žádná

typová kontrola, syntaxe je opravdu velmi volná a hlavně zde nejdou definovat vlastní datové typy.

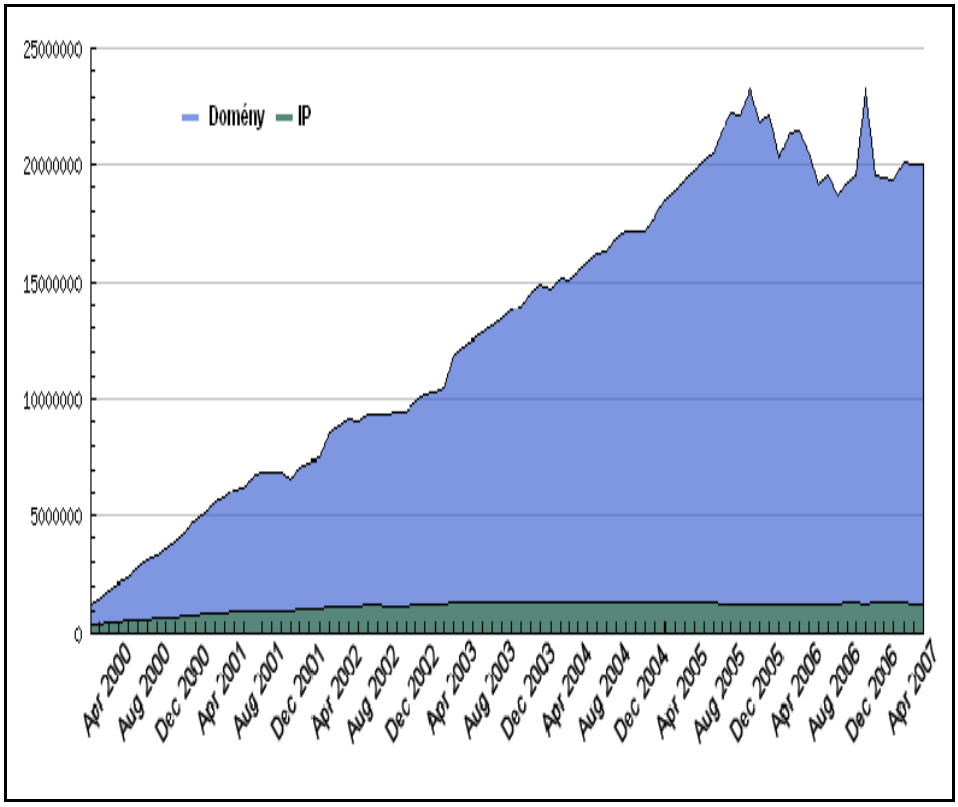
5. 1. 3. Python

je interpretovaný objektově orientovaný programovací jazyk navržený v roce 1990 Guido van Rossumem, který umožňuje psát i rozsáhlé plnohodnotné projekty, aniž by se v něm programátor ztratil (jazyk je poměrně krátký a dobře čitelný). Nenásilně řeší otázku objektově orientovaného programování. Spadá také pod GNU GPL Actistic licencí a též je multiplatformní. Je vhodný pro psaní prototypů aplikací, webových aplikací, aplikací typu klient/server, síťových aplikací, skriptů, utilit, systémových programů, menších GUI aplikací, práci s textovými dokumenty včetně XML, skriptování jiných aplikací a podobně.

5. 1. 4. PHP

… někdy v roce 1994, kdy Rasmus Lendorf dal dohromady kombinaci skriptů v Perlu, aby zjistil, kdo se díval na jeho výsledky. Pozvolna se lidé začali o tyto skripty zajímat, takže byly později vydány jako balíček „Personal Home Page“ tools (původní význam PHP). Vzhledem k zájmu napsal skriptovací jádro (asi v roce 1995) společně s jiným nástrojem pro analýzu vstupu z formulářů v HTML: FI (Form Interpreter), tedy to, co bylo nazýváno PHP/FI nebo PHP2. Poté skupina programátorů (Lersdorf, Gutmans, Suraski, Bakken, Caraveo, a Winstead) zlepšila a rozšířila skriptovací jádro a přidala jednoduché API, které umožňuje jiným programátorům volnost v přidávání funkcí do jazyka vytvořením modulů (to je PHP3). PHP4 je založená na jádře Zend – to je navrženo tak, aby bylo jednoduše implementovatelné do různých aplikací. Nejnovější verzí je PHP5, která obsahuje mnoho změn, ale tou nejvýraznější je nepochybně změna objektového modelu.

PHP je možná nejrozšířenějším skriptovacím jazykem používaném pro tvorbu internetových aplikací, což dokazuje i graf z obrázku č. 4, kde vidíme, že v současné době využívá tohoto jazyku 25 000 000 domén.



Obrázek č. 4 – Využívání PHP na serverech (3)

6. Databáze

V dnešní době se používá SQL, tj. databázové jádro, které implementuje standardní specifikaci SQL (Structured Query Language) (verze implementovaná do nejnovějších databází je známá jako SQL2, SQL-92 nebo prostě SQL). V relačních databázích jsou data uložena v sadě tabulek. Každá tabulka obsahuje jeden nebo více sloupců, které popisují atributy dat a každý řádek v tabulce je instancí dat.

Relační systémy řízení báze dat (Relational Database Management System – RDMS) se používali mnoho let akademicky i obchodně (některé se začaly používat v podstatě ve vývojových laboratořích) a jsou ověřenými a stabilními systémy se silným zázemím metod a technik, které pomáhají vývojářům navrhovat a vytvářet databázové aplikace, které se hodí pro jejich potřeby.

6. 1. Dělení databází

To je kupodivu čím dál tím těžší, protože jednotlivá kritéria se v poslední době vzájemně překrývají a mnoho databází umí hodně podobných věcí. Pokusím se alespoň ukázat, jak různě se dají databáze dělit.

⇒ **Objektové a relační** – databáze se liší podle způsobu ukládání dat. Ve světě je rozšířenější relační model.

⇒ **Jednouživatelské a víceuživatelské** – dělíme podle toho, kolik uživatelů se k databázi může připojit. Pochopitelně „v reálném světě“ jednouživatelským databázím prakticky odzvonilo, nýbrž je užitečné vědět, že i víceuživatelskou databázi lze většinou nakonfigurovat jako jednouživatelskou a že to může mít své opodstatnění.

⇒ **Souborové a systémové** – buď databáze používá pro uložení dat jeden soubor (např. dbf), nebo je úložiště dat zabudováno v systému. U souborových databází stačí výše zmíněný soubor přenést na jiný stroj a je okamžitě znovu použitelný. U systémových databází se záloha a obnova dělá nějak jinak (většinou je to exportem posloupnosti příkazů, jež nám později vytvoří ekvivalentní databázi, do textového souboru). V posledních letech ale toto dělení trochu ztratilo význam, jelikož je většina databází systémových.

⇒ **Podle licence a ceny** - kód databáze může být uzavřený nebo otevřený. Šíření software může být svobodné nebo může podléhat nějakým podmínkám. Databáze se může využívat bez poplatků nebo může jít o placený software. Jsou i databáze jež jsou open source, ale pro použití při komerčních účelech si ji musíme zaplatit.

⇒ **Podle toho, kde běží** - na jednoplatformní a multiplatformní. Jednoplatformní poběží pouze na některém systému (třeba na Windows), multiplatformní na více systémech (GNU/Linux, Microsoft Windows, FreeBSD, Sun Solaris, IBM's AIX, Mac OS

X, HP-UX, AIX, QNX, Novell NetWare, SCO OpenUnix, SGI Irix, and Dec OSF.).

⇒ **Podle velikosti, výkonu a vhodnosti nasazení** - Databáze mívají limity ve velikosti, počtu současně přihlášených uživatelů, počtu současně probíhajících procesů a podobně. Obecně je totiž obtížné nějak rovnoprávně posoudit databáze.

6. 2. Příklady databází

6. 2. 1. MySQL

je velmi populární databáze. Mnoho zdrojů rovněž uvádí, ze je to velmi rychlá databáze. Neoplývá však tolika funkcemi a možnostmi jako některé její konkurenční databázové systémy. MySQL se nachází někde uprostřed pomyslného žebříčku vhodnosti nasazení a že v malých až středních projektech ji rozhodně použít můžete. Samozřejmě má své zastánce a odpůrce. Odpůrci dost často tvrdí, že MySQL je tak rozšířená jen proto, že webhostingové společnosti ji často nabízejí pro hostované weby jako součást portfolia svých služeb. Kdyby tomu tak nebylo a tyto společnosti se rozhodly upřednostnit jiný software, tak by popularita MySQL přinejmenším hodně klesla.

Naproti tomu zastánci tvrdí, že tyto společnosti nabízejí MySQL proto, že je dobrá, a kdyby byla na trhu lepší databáze než MySQL, byla by nabízena. Zastánci MySQL prostě tvrdí, že si ji spotřebitelé vybrali a to jasně ukazuje na její kvality.

Pravda je asi někde uprostřed těchto dvou tvrzení. MySQL bude určitě ještě nějaký čas hodně rozšířená, takže se vyplatí ji používat.

Abych se vrátil k mému rozdělení tak MySQL je relační víceuživatelská multiplatformní databáze se systémovým ukládáním dat. Licence MySQL je duální. Více bychom se mohli dozvědět na http://www.mysql.com/company/legal/licencing. Jednoduše řečeno - pokud budeme vyvíjet software pod GPL kompatibilní licencí, můžete použít MySQL pod licencí GPL.

6. 2. 2. PostgreSQL

Autory první verze PostgreSQL 1.01 byli Andrew Yu and Jolly Chen. Do portace, testování, ladění a rozšiřování kódu se zapojilo mnoho dalších vývojářů. Původní kód Postgresu, ze kterého PostgreSQL vychází, je výsledkem úsilí mnoha studentů a programátorů pracujících pod vedením prof. Michaela Stonebrakera na University of California v Berkley. Původní název software z Berkley byl Postgres. Po přidání jazyka SQL se název změnil na Postgres95. Koncem roku 1996 byl RDBMS přejmenován na PostgreSQL. PostgreSQL je šířen pod BSD licencí, která je nejliberálnější ze všech open source licencí. Tato licence umožňuje neomezené používání, modifikaci a distribuci PostgreSQL. Můžete ji šířit se zdrojovými kódy nebo bez nich, zdarma nebo komerčně. Splňuje kritéria ACID, z SQL podporuje cizí klíče, JOIN tabulek, pohledy, spouště, vnořené příkazy SELECT, příkazy CASE, COALESCE a NULLIF. Uživatelem definované funkce (UDF) mohou být psané v několika programovacích jazycích. UDF lze využít k návrhu vlastních datových typů, konverzních a agregačních funkcí. PostgreSQL umožňuje běh uložených procedur napsaných v několika programovacích jazycích, v Perlu, v Python, v jazyku C nebo v speciálním PL/pgSQL, jazyku vycházejícím z PL/SQL fy. Oracle. Obsahuje kvalitní, přehlednou a dobře udržovanou dokumentaci. Současným koordinátorem všech vývojářů vyvíjecích PostgreSQL je Marc G. Fournier.

PostgreSQL Běží na všech rozšířených operačních systémech včetně Linuxu, UNIXů (AIX, BSD, HP-UX, SGI-IRIX, Mac OS X, Solaris, Tru64) a Windows. V instalačních instrukcích naleznete aktuální seznam všech platforem, na kterých byla testováním ověřena funkcionalita PostgreSQL. Poslední verze PostgreSQL je 7.4.3.

6. 2. 3. Oracle

je systém řízení báze dat (Oracle database manag ement system – DBMS), moderní multiplatformní databázový systém s velice rozsáhlými možnostmi zpracování dat, velmi vysokým výkonem a snadnou uspořádatelností.

Databázový systém Oracle je vyvíjen firmou Oracle Corporation. To je jedna z největších a nejdůležitějších společností vyvíjejících relační databáze, nástroje pro vývoj a správu databází či customer relationship management (zkráceně CRM) systémů. Společnost byla založena v roce 1977 a v roce 2005 zaměstnávala 50 000 lidí. Má zastoupení ve 145 zemích světa. Výkonný ředitel firmy je Lawrence J. Ellison, jenž je považován za jednoho z nejbohatších lidí na planetě.

Aktuální verze je Oracle Database 10g. Tento systém podporuje nejen standardní relační dotazovací jazyk SQL podle normy SQL92, ale také proprietární firemní rozšíření Oracle (např. pro hierarchické dotazy), imperativní programovací jazyk PL/SQL rozšiřující možnosti vlastního SQL (v tomto jazyce je možné tvořit uložené procedury, uživatelské funkce, programové balíky a triggery), dále podporuje objektové databáze a databáze uložené v hierarchickém modelu dat (XML databáze, jazyk XSQL).

7. Bezpečnost

Nedílnou součástí internetového obchodování je také bezpečnost při provádění plateb přes internet. K tomu nám slouží následující protokoly.

7. 1. SSL

Secure Sockets Layer – je protokol, který je vložen mezi transportní vrstvu (např. TCP/IP) a vrstvu aplikační (např. HTTP). Po vytvoření SSL spojení je komunikace mezi klientem a serverem šifrovaná => bezpečná.

Ustavení SSL spojení funguje na principu asymetrické šifry, kdy každá z komunikujících stran má dvojici šifrovacích klíčů - veřejný a soukromý. Veřejný klíč je možné zveřejnit a pokud tímto klíčem kdokoliv zašifruje nějakou zprávu, je zajištěno, že ji bude moci rozšifrovat jen majitel použitého veřejného klíče svým soukromým klíčem. Nejčastěji se tento protokol využívá pro zabezpečenou komunikaci s internetovými servery pomocí HTTPS.

7. 2. HTTPS

Je nadstavba protokolu HTTP, která zajišťuje zvýšenou bezpečnost před odposloucháváním nebo podvržením dat. Data jsou stále přenášena pomocí HTTP. Není to už běžný text, ale jsou to data šifrována pomocí SSL. Komunikace probíhá na TCP portu 443.

8. Dostupná řešení internetových obchodů

Když si chceme založit internetový obchod, tak toho můžeme dosáhnout třemi způsoby, které mají vždy své pro a proti.

8. 1. 1. Vlastní vývoj

Návrhem vlastního internetového obchodu získáme produkt „šitý na míru“ prakticky za nulové náklady, ale programátora to stojí spoustu času. Dále se musí zvážit spousta aspektů než se vůbec do samotného programování pustí. Nejdříve zhodnotit, jak velký (tím myslím objem sortimentu) má obchod být a podle toho zvolit i případné technologie, které jsem uvažoval v předchozích kapitolách. Navíc musí mít obchod perfektní návrh, aby se při případných změnách nestal nepoužitelným, což v následujících variantách je většinou vyřešené.

8. 1. 2. E-shopy volně ke stažení

Dnes již existuje mnoho již předprogramovaných e-shopů, které jsou na internetu volně ke stažení a jejich používání je legální i při komerčním využití. Drtivá většina těchto obchodů je open-source. Dost často se může stát, že potřebujeme z různých důvodů dělat úpravy, ale než se začneme orientovat v kódu, tak to dost dlouho trvá a navíc, když se někde v kódu objeví bezpečnostní chyba, tak ji poté může někdo lehce zneužít, jelikož do nich může úplně každý.

K nejvyužívanějším e-shopům z tohoto oboru patří *VirtueMart*, který ve spojení s redakčním systémem Joomla patří k nejlepším ve svém oboru, jelikož se do jeho vývoje velice často zapojují i samotní uživatelé a technická podpora je na vysoké úrovni.

8. 1. 3. Komerční řešení

Toto řešení je sice nejdražší, ale za to bychom měli dostat e-shop bezpečný, s kompletní dokumentací a technickou podporou. Bohužel většina firem (samozřejmě ne všechny), programujících podobné projekty neposkytuje zdrojový kód, což znamená, že svůj „obchod" dostaneme nějakým způsobem zašifrovaný, tudíž o každou menší úpravu musíme žádat výrobce a opět za ni platíme.

Opět uvedu příklad: *Zoner inShop 3* je nejpoužívanějším komerčním řešením pro výstavbu a provoz profesionálních internetových obchodů v ČR. Tento systém disponuje širokou škálou základních, pokročilých i vysoce profesionálních prodejních a marketingových funkcí a zároveň maximálními možnostmi nastavení a přizpůsobení podle potřeb obchodníka či podle specifik jeho produktů.

9. Zhodnocení úvodní části

Při programování budu používat asi tu nejběžnější kombinaci – jako webový server použiji Apache HTTP server, na programování použiji PHP verze 5, se kterou je nejlepší pracovat s databází MySQL. Následně udělám kombinaci mezi „vlastním vývojem" a „e-shopy volně ke stažení", jelikož nejdříve ceníky z několika zdrojů nahraji pomocí vlastní aplikace do své databáze, ze které to poté přehraji do redakčního systému Joomla, do kterého si importuji obchod VirtueMart, což znamená, že interface projektu bude tvořit open-source řešení.

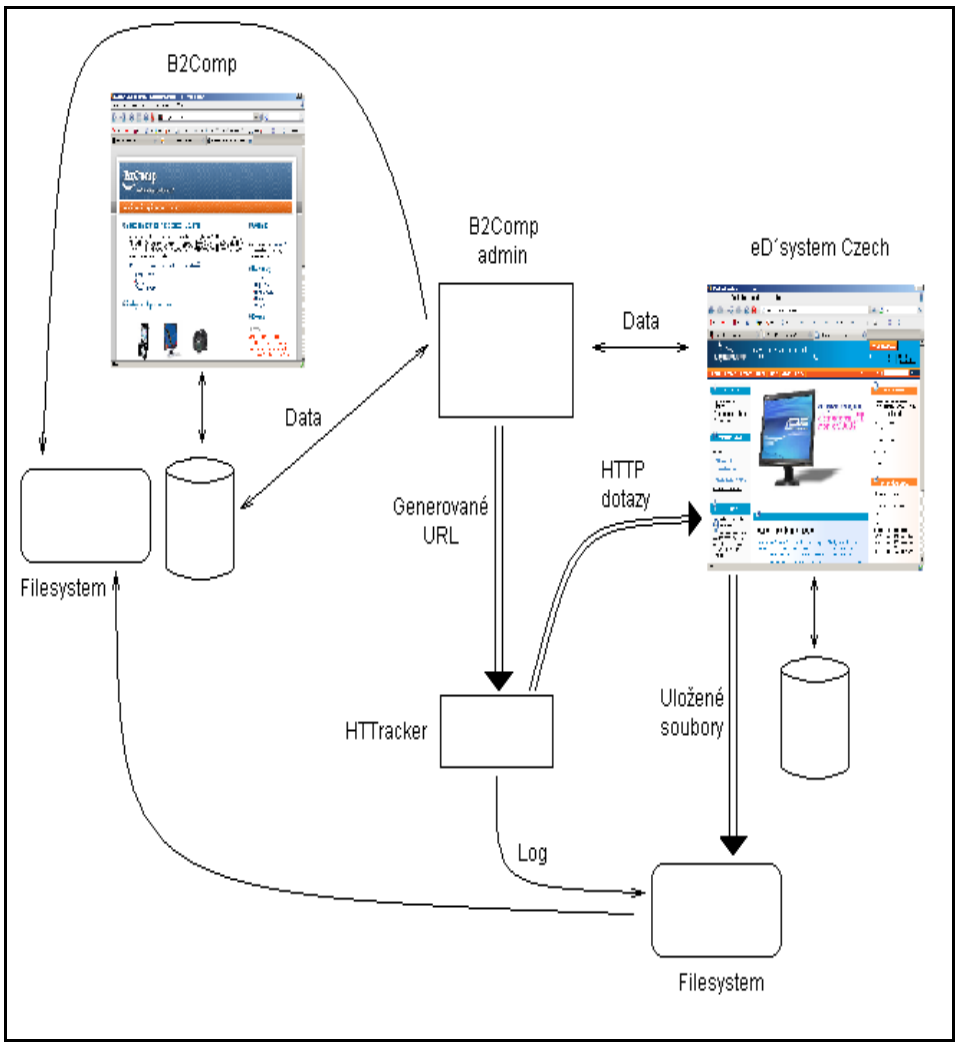
10. Řešení problematiky e-shopu společnosti B2Comp

Nyní již přichází na řadu vlastní popis problematiky návrhu databáze v konkrétním projektu, v mém případě internetového obchodu, zabývajícím se prodejem kancelářských potřeb, kalkulaček, počítačů, PC komponentů, tiskáren, notebooků, MP3 přehrávačů a elektronických překladačů.

Tento obchod provozuje společnost B2Comp sídlícího v Praze již od roku 2000. Společnost se prezentovala i na několika výstavách –

z těch nejznámějších bych jmenoval Invex, kde se tato firma prezentovala v letech 2004, 2005 a 2006, a na Schola Nova v roce 2004 obdrželi ocenění za nejlepší exponát.

10. 1. Původní návrh projektu



Obrázek č. 5 – Původní návrh projektu

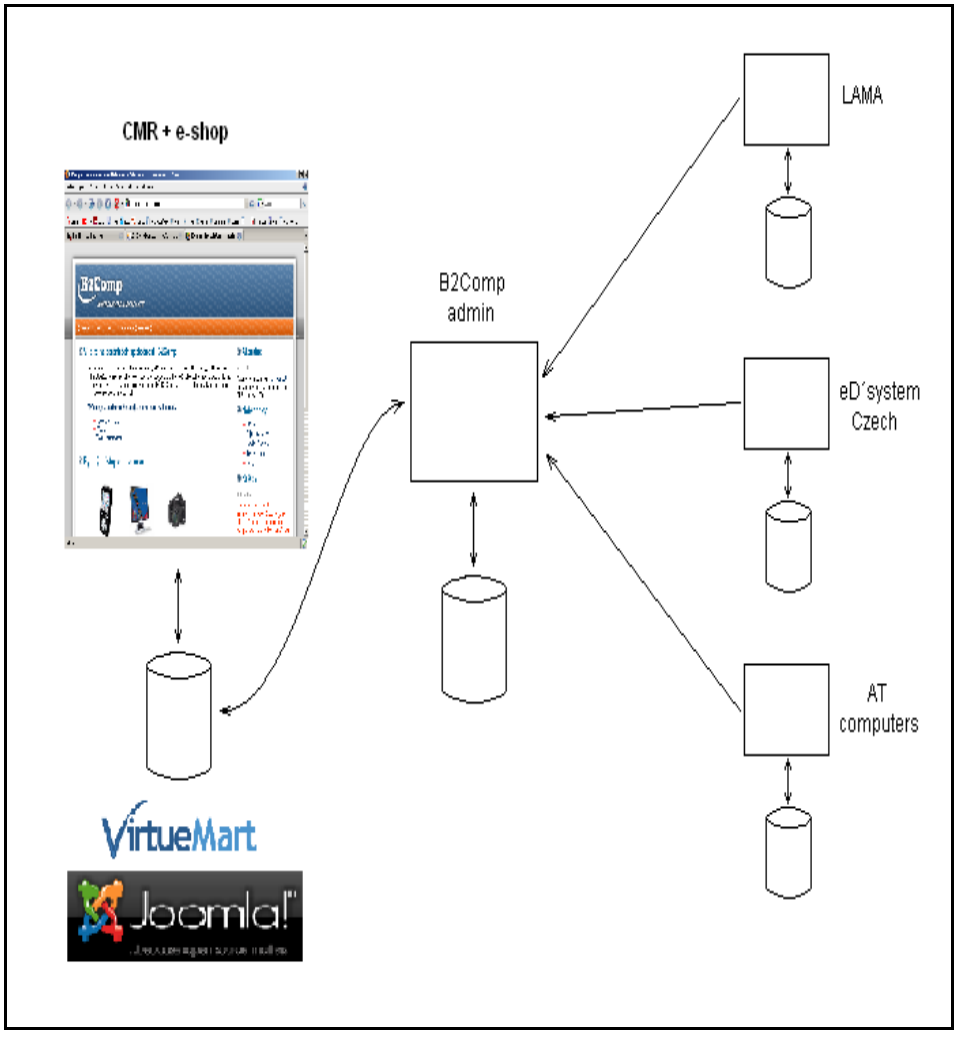
Na obrázku č. 5 vidíme původní návrh projektu internetového obchodu fungujícího na adrese http://www.kalkulacky.cz. Data do ceníku čerpá z e-shopu společnosti eD´system Czech, který je na adrese http://www.edcz.cz. Tento obchod poskytuje svoji databázi v podobě několika *.mdb souborů (to je databáze vytvořená v programu Microsoft Access). Všechna tyto data se přes PHP skripty nahrála do stejně navržené databáze jako má obchod eD´system Czech.

V tomto okamžiku vyvstal problém s obrázky, které v jimi poskytované databázi nebyly. K tomu se použil HTTracker, který na

jejich obchod posílal HTTP dotazy a stahoval obrázky do dočasného filesystému a přitom zaznamenával do „log“ souboru tyto dotazy a obrázky, které mu server na každý dotaz vracel. Poté se všechno nahrálo do filesystému na server B2Comp. Obrázky se následně generovali z výše zmíněného „log“ souboru.

Další problém vznikl, když se mělo přidat zboží od jiného dodavatele. V tomto případě museli zaměstnanci obchodu ručně nahrávat každý výrobek zvlášť, což je pracné a taky velice časově náročné. A navíc přidání ceníku od dalšího dodavatele by bylo v tuto chvíli strašně složité, kvůli četnosti vazeb.

10. 2. Nový návrh projektu



Obrázek č. 6 – Nový návrh projektu

Na obrázku č. 6 je vidět návrh projektu, který nám řeší hned několik problémů, které vyvstali předchozím řešením. Základním

stavebním kamenem je aplikace *B2Comp admin*, kde je navržená vlastní databáze. Tato aplikace řeší problém velké četnosti vazeb, což znamená, že je v podstatě jedno, z kolika obchodů nebo různých databází budeme chtít nahrávat produkty do té naší. Vždy existuje mezi touto aplikací pouze jedna vazba, což jí dělá velice flexibilní.

Interface našeho projektu bude zajišťovat výše zmiňovaná open-source kombinace CMR (redakčního systému) Joomla (http://www.joomla.org) a e-shopu VirtueMart (http://www.virtuemart.net). Převod dat do této databáze je zajištěn opět jednou vazbou z aplikace *B2Comp admin*. Toto všechno dělá tento návrh velice flexibilním.

11. Integrace dat

Motivačním prostředkem pro integraci dat je v našem případě úspora lidských zdrojů. Dosavadní roztříštěnost systému vedla k velké pracnosti s obsluhou, což byla monotónní práce, kde je navíc tendence vysoké chybovosti a dalších nežádoucích efektů

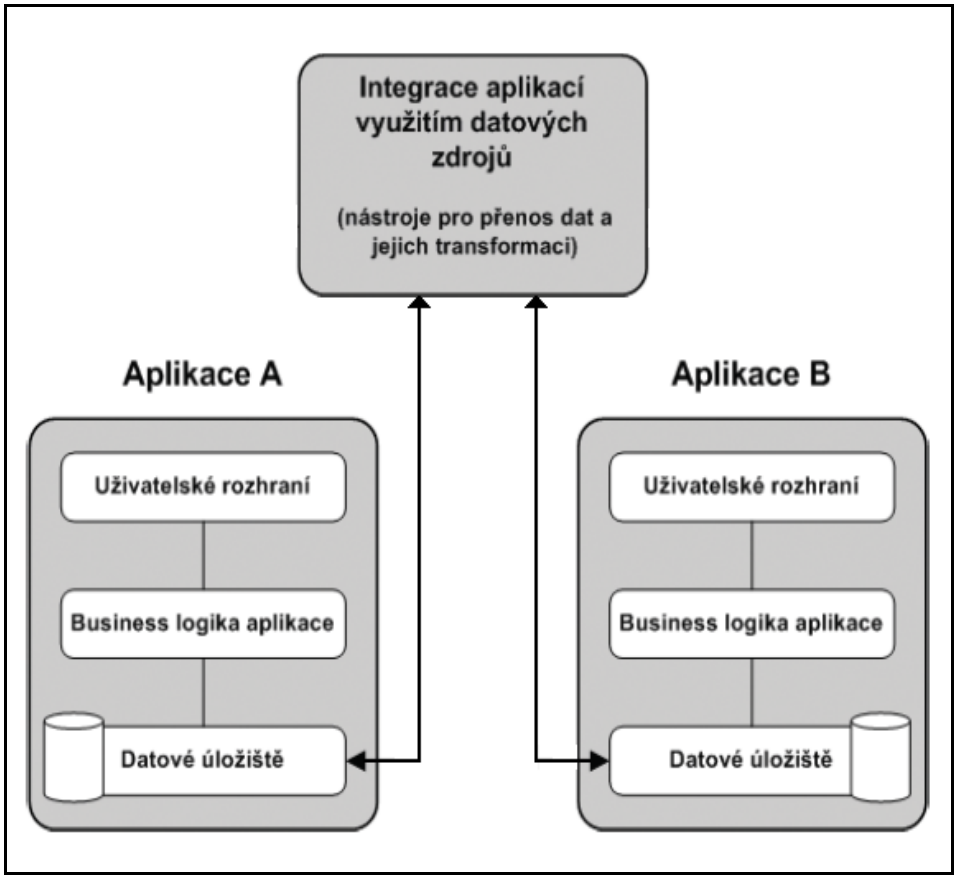
11. 1. Integrace na datové úrovni

„Jeden ze zdánlivě dobrých způsobů integrace spočívá ve své podstatě v ignorování existence aplikací. Většina jich totiž funguje víceméně stejně – výsledkem jejich činnosti jsou data uložená v relační databázi, přičemž všechny běžně používané relační databáze fungují na víceméně stejných principech. Vzhledem k tomu, že většina aplikací se bez prostudování dokumentace jeví jako černé skříňky (mnohé i po prostudování dokumentace), je obcházení aplikace a přímá práce s daty velmi lákavá (viz obrázek č. 7). Většinou se přitom používají datové pumpy (ETL, extraction-transformation-loading), databázové replikace, export/import a podobné postupy.

Jde o docela dobré a rychlé taktické řešení, které poměrně rychle a bezbolestně přinese dílčí úspěchy. Problém spočívá ve velmi omezené škálovatelnosti tohoto řešení. V okamžiku, kdy počet různých datových zdrojů přesáhne číslo dva anebo je v rámci zdroje nutné pracovat s větším množstvím tabulek (což lze obojí považovat téměř za jisté),

vzniká velmi neprůhledná a špatně udržovatelná struktura. Dalším problémem je možnost konfliktů – pokud určitá data nemají autoritativní (někdy nazývaný též kanonický) zdroj, je nutné vymýšlet složitou logiku pro řešení možných konfliktů aktualizace dat. Navíc přechod na novou verzi aplikace často znamená radikální změny ve schématu databáze, čímž může být dosavadní vložená práce výrazně znehodnocena. Často se též nelze vyhnout zabudování částí aplikační logiky do transformačního procesu, což je ošidné z hlediska přehlednosti, koncepčnosti a udržovatelnosti řešení. Tím nechci říct, že tento přístup je vždy špatný – hodí se velmi dobře například pro vytváření operačních datových skladů, kde je velký objem dat, je jasný jejich primární zdroj a můžeme si dovolit nezbytnou neaktuálnost (latenci) dat v datovém skladu. Pokud ale chceme mít integrační řešení flexibilní, koncepční a pracující v téměř reálném čase, nelze integraci s využitím datových zdrojů v žádném případě doporučit.“

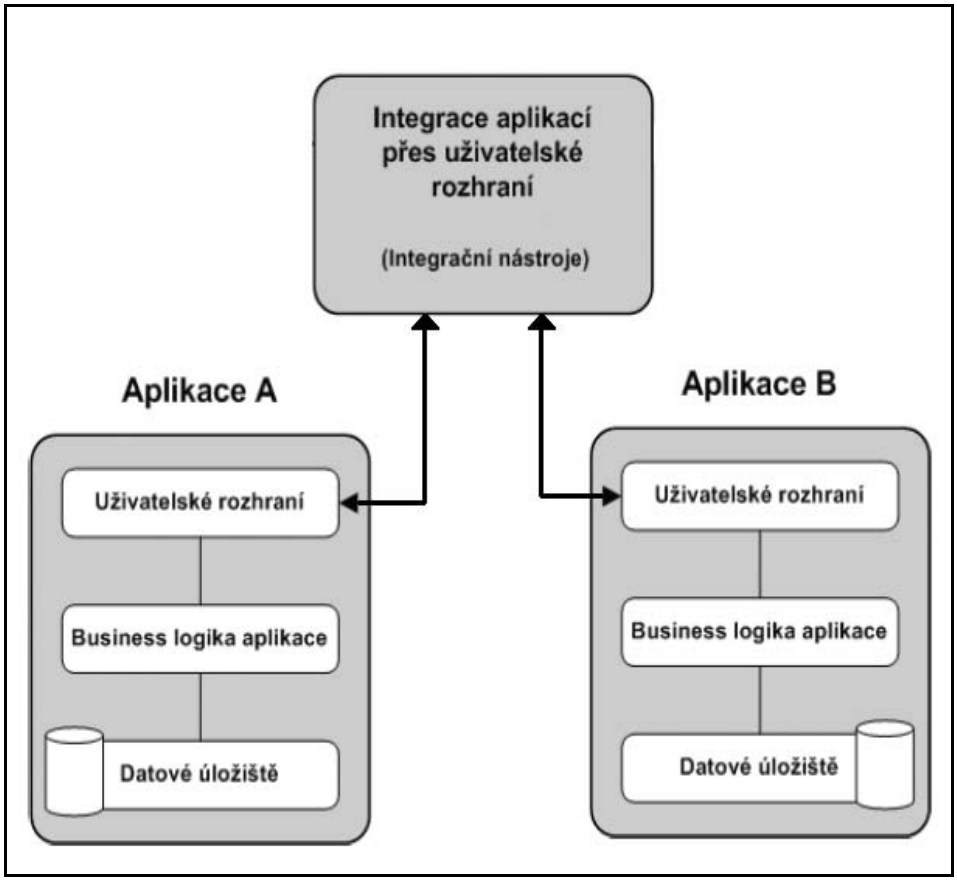
JUŘEK, Michael, Moderní integrace aplikací, Praha: Microsoft, s. r. o.



Obrázek č. 7 – Integrace na datové úrovni (4)

11. 2. Integrace na úrovni uživatelského rozhraní

„Integrace na úrovni uživatelského rozhraní je přesně opačným extrémem. Zde aplikace naopak respektujeme jako nebezpečné černé skříňky a hledáme si co nejdelší „klacek“, kterým bychom se jich dotýkali, a tím je klávesnice a obrazovka. Při integraci na úrovni uživatelského rozhraní de facto nahrazujeme koncového uživatele obsluhujícího aplikaci automatizovaným postupem, který čeká na nápovědu od aplikace a reaguje na ni simulací uživatelského vstupu – v anglicky psané literatuře se tento postup nazývá screen scraping (viz obrázek č. 8).



Obrázek č. 8 – Integrace na úrovni uživatelského rozhraní (4)

V určitých situacích je tento postup výhodný. Pokud je aplikace již velmi stará, není k dispozici zdrojový kód a dodavatel aplikace již dávno zanikl, jde často o jediný možný postup jak eliminovat nákladnou lidskou práci potřebnou při předávání údajů mezi aplikacemi (obzvláště, jde-li o pokročilého uživatele schopného zvládnout operaci Kopírovat/Vložit). V drtivé většině situací je ale tento postup naprosto

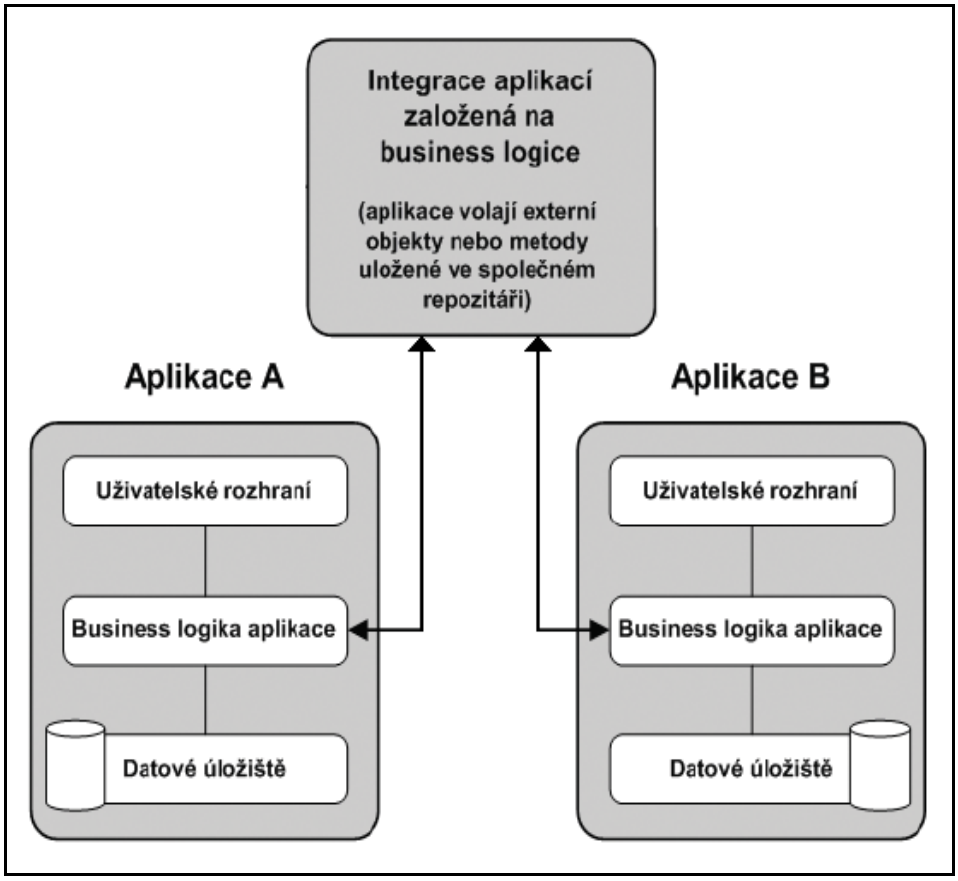
neadekvátní – jde opravdu pouze o krátkodobé nekoncepční řešení, které je kriticky závislé na neměnnosti aplikace, tedy něčem, co lze jenom těžko předpokládat.

Jinou kapitolou je dnes moderní integrace uživatelského rozhraní v portálech. Zde se ovšem jedná spíše o sjednocení uživatelského rozhraní a jeho přiblížení koncovému uživateli, nikoliv o výměnu dat mezi integrovanými aplikacemi. Jde jistě o moderní a racionální trend, které s popisovanou problematikou úzce souvisí, ale není teď pro nás tím hlavním.“

JUŘEK, Michael, Moderní integrace aplikací, Praha: Microsoft, s. r. o.

11. 3. Integrace na úrovni obchodní logiky

„Střední cestou, která ovšem v tomto případě není zlatá, je integrace na bázi komponent obchodní logiky. Spočívá v integraci prostřednictvím vzájemného volání mezi objekty (komponentami) ve střední vrstvě aplikací (viz obrázek č. 9), jde v podstatě o „drátování“ jednotlivých aplikací mezi sebou.



Obrázek č. 9 – Integrace na úrovni obchodní logiky (4)

Ačkoliv tento přístup na první pohled vypadá poměrně koncepčně, je jeho praktická realizace z mnoha důvodů velmi obtížná:

⇒ vyžaduje si zásah do aplikací, což s sebou často přináší nastudování velkého množství dokumentace a prozkoumávání bílých míst, složitou koordinaci s autorem či dodavatelem aplikace, problémy s podporou vzniklého řešení a další;

⇒ vzájemná závislost a těsná vazba aplikací. Vzhledem k tomu, že dochází ke vzájemnému volání aplikací, stává se dostupnost a správná funkce volající aplikace do velké míry závislou na všech volaných aplikacích – tento nedostatek lze řešit používáním asynchronního volání, což je ale velmi pracné a často pouze jednoúčelové a komplexní řešení, které musí být podporováno řadou servisních aplikací, které celý systém komplikují. Vzájemná závislost a těsná vazba jsou též brzdou aktualizace aplikací – změna v aplikaci si vyžaduje analýzu dopadu změn v ostatních aplikacích, implementaci změn a znovuotestování všech závislých aplikací;

⇒ syndrom N^2 – integrace na úrovni obchodní logiky vede k růstu složitosti s počtem možných vazeb mezi aplikacemi, přičemž tento počet roste s kvadrátem počtu integrovaných systémů. Přidání každé další integrované aplikace je tak vždy pracnější a nákladnější než té předchozí – rozhodně ne dobrá strategická vyhlídka.“

JUŘEK, Michael, Moderní integrace aplikací, Praha: Microsoft, s. r. o.

11. 4. Zhodnocení integrace

V našem případě použijeme integraci na datové úrovni hned z několika důvodů. Všechny tři výše zmiňované obchody nám poskytují v pravidelných intervalech své vyexportované databáze, které můžeme zkonvertovat do té naší, čímž z výběru vyřadíme integraci na uživatelské úrovni. Jak jsem již zmiňoval výše, tak chceme abychom kdykoli mohli přidat další a další ceníky, kvůli čemu jsme se chtěli zbavit četnosti vazeb, které by nám při použití oné zlaté střední cesty (integrace na úrovni obchodní logiky) přibývaly již zmiňovaným N^2 efektem. Int. na

datové úrovni by pro nás měla nevýhodu, kdybychom potřebovali zajistit neustálou aktuálnost dat, ale jelikož se ceny mění řekněme v pravidelných intervalech (přibližně 1 týden), tak nás to nijak zvlášť zajímat nemusí.

12. Řešení jednotlivých problémů

V této fázi se zaměřím na samotné zpracování nového návrhu a jeho implementaci. Bude k tomu použit webový server Apache 2.2.3, jako skriptovací jazyk jsem zvolil PHP verze 5.2.0 a mezi databázemi jsem vybral MySQL ve verzi 5.0.27. K přístupu k databázi používám aplikaci naprogramovanou taktéž v PHP phpMyAdmin 2.9.1.1.

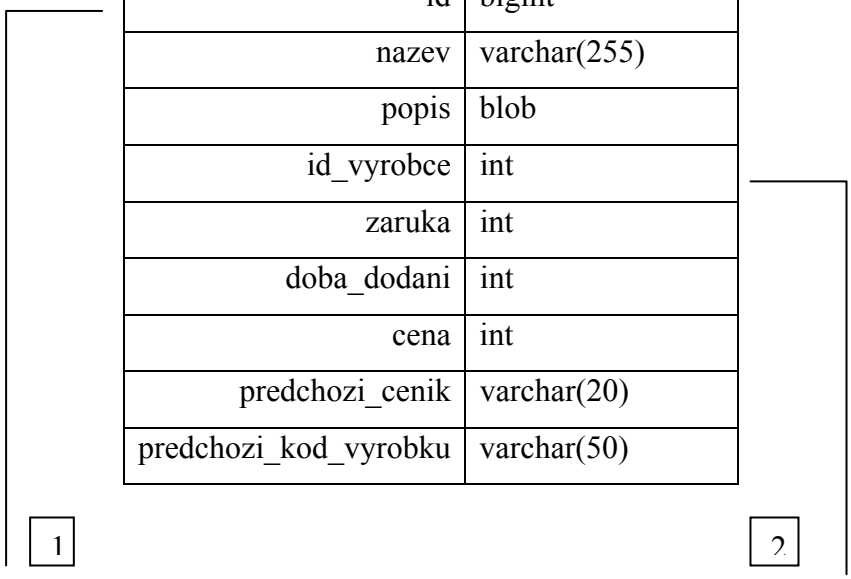
12. 1. Návrh databáze

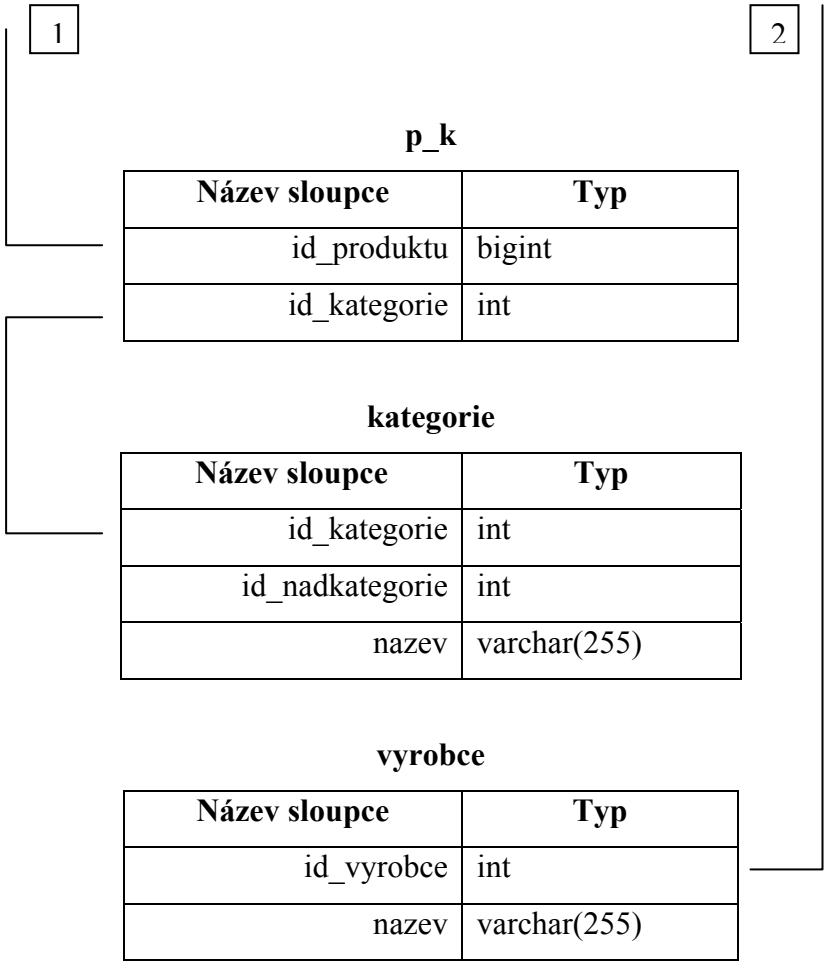
Zde se zaměřím na jednu z nejdůležitějších částí celého projektu. Při špatném návrhu databáze by se mohl vyskytnout problém buď s nahráváním ceníků od různých návrhářů, nebo na druhou stranu při exportování databáze do již připravené databáze e-shopu VirtueMart, do níž je nemožné jakkoli zasahovat.

12. 1. 1. Jednotlivé tabulky

Produkt

Název sloupce	Typ
id	bigint
nazev	varchar(255)
popis	blob
id_vyrobce	int
zaruka	int
doba_dodani	int
cena	int
predchozi_cenik	varchar(20)
predchozi_kod_vyrobku	varchar(50)





12. 1. 2. Popis

Základem je tabulka *produkt*, kde se uchovávají všechny informace o prodávaných produktech, včetně názvu ceníku, ze kterého pochází a jeho bývalého kódu, pod kterým se tam nacházel, kvůli snazší zpětné dohledatelnosti. Další tabulka *p_k* řeší vazbu produktů na jednotlivé kategorie. Mohl jsem jednoduše přidat do tabulky *produkt* sloupec *id_výrobce*, ale některé výrobky mohou spadat do několika kategorií. Tato tabulka se dále odkazuje na tabulku *vyrobce*, která obsahuje ručně naplněnou tabulku výrobců. Poslední tabulka je *kategorie*. Na ní se odkazuje opět tabulka *produkt* a její sloupec id_kategorie. Také řeší problém víceúrovňového rozvrstvení.

12. 2. Výrobci

Jelikož jsou ceníky od tří různých dodavatelů, jsou názvy výrobců u každého jinak. Např. firma Hewlett-Packard může být uváděna jako „HP“, „H-P“ nebo „Hewlett-Packard“. Původní jsem to chtěl udělat

spárováním v samotné databázi, ale špatně navržená struktura ceníku od firmy Lama to neumožnila, tím pádem jsem to udělal funkcí s jedním parametrem. Funkci předávám kód výrobce, jak ho mají uváděného v ceníku a ta nám pak podle kódu vrací id_výrobce, odpovídajícího naší databázi.

```
function dej_id_vyrobce_lama ($vyrobce){
    switch ($vyrobce) {
        case "3M": return 1; break;
        case "AT": return 3; break;
        case "AE": return 9; break;
        case "BA": return 15; break;
        case "BR": return 18; break;
        case "CA": return 19; break;
        case "CI": return 21; break;
        case "DW": return 28; break;
        case "DL": return 29; break;
        case "DE": return 31; break;
        case "EP": return 34; break;
        case "EU": return 35; break;
        case "FJ": return 37; break;
        case "FS": return 38; break;
        case "FU": return 39; break;
        case "GN": return 40; break;
        case "GE": return 41; break;
```

```
function dej_id_vyrobce_edcz ($vyrobce){
    switch ($vyrobce) {
        case "3CM": return 2; break;
        case "ACP": return 4; break;
        case "APL": return 5; break;
        case "ADA": return 6; break;
        case "ALI": return 7; break;
        case "AMD": return 8; break;
        case "APA": return 9; break;
        case "APC": return 10; break;
        case "ASU": return 11; break;
        case "ASL": return 12; break;
        case "ASR": return 13; break;
        case "ATI": return 14; break;
        case "BLK": return 16; break;
        case "BEN": return 17; break;
        case "CAN": return 19; break;
        case "CDI": return 19; break;
        case "CPR": return 20; break;
```

Obrázek č. 10 – Ukázky zdrojových kódů 1

Pokud se objeví nějaký nový výrobce, který ještě není v naší databázi, vrací tato funkce číslo 0, takže se dají tyto funkce jednoduše rozšířit.

12. 3. Kategorie

Podobný problém jako u výrobců vznikl i u kategorií s tím rozdílem, že v jednom ceníku jsou CD, DVD a další přenosná média uvedená jako „Datová média“ a jinde jako „CD, DVD, Diskety, Blue-ray disky atd...“. Tím bohužel vznikl problém, že jsem například musel všechna datová média vkládat do jedné kategorie a nemohl jsem je roztřídit do podkategorií, jelikož kdyby zákazník klikl na podkategorii, tak by se mu zobrazili produkty pouze z ceníků, kde to takhle bylo roztříděno. Stejně jako u výrobců se dá tato funkce rozšířit, když by požadovaná kategorie ještě neexistovala. V tomto případě taktéž vrací 0.

```
function dej_id_kategorie_lama ($kategorie){
    switch ($kategorie){
        case "KOA": return 200;
        case "SCS": return 108;
        case "VGL": return 200;
        case "TCN": return 200;
        case "ARC": return 200;
•
•
•

function dej_id_kategorie_lama ($prvni, $druha){
    switch ($prvni){
        case "Pamětová média":{
            switch($druha){
                case "Compact Flash card": return 211;
                case "Multimedia Memory card": return 212;
                case "Secure Digital Card": return 213;
                case "Memory Stick": return 214;
                case "XD Picture Card": return 215;
                case "USB Card Reader": return 216;
                case "USB stick": return 217;
                case "USB příslušenství": return 218;
            }
        }
•
•
•
```

Obrázek č. 11 – Ukázky zdrojových kódů 2

12. 4. Nahrávání ceníků

Každý z dodavatelů poskytuje ceník v jiném formátu. Proto musel být u každého zvolen trochu jiný způsob nahrávání do databáze SQL, ale konečný algoritmus je stejný. Největší problém zde byl s názvy produktů, kde v každém ceníku se mohou vyskytovat stejné produkty, ale pokaždé budou mít trochu odlišný název. Tuto problematiku řeší funkce *porovnej_nazvy($nazev1, $nazev2)*, která rozdělí řetězce *$nazev1* a *$nazev2* do polí podle dvou zadaných znaků, v našem případě „ “ a „/“(obrázek č. 12) a porovná je mezi sebou. Funkce vrací *true*, když je shodnost vyšší než 80 %.

```
$dily = explode(" ", $nazevl);
$pocet = 0;
foreach ($dily as $v){
    if (ereg("/", $v)){
        $tmp = explode("/", $v);
        $pom = count($tmp);
        for ($i=0; $i<$pom; $i++){
            $nl[$pocet++]=$tmp[$i];
        }
    } else $nl[$pocet++]=$v;
}
```

Obrázek č. 12 – Ukázky zdrojových kódů 3

12. 4. 1. Lama

Společnost LAMA Plus s. r. o. dodává svůj ceník v souborovém formátu CSV („Comma Separated Values“). Ten je možno otevřít hned v několika aplikacích – Microsoft Excel, OpenOffice Calc nebo KSpread. Na načítání dat z tohoto formátu existuje v PHP funkce:

fgetcsv ($fp, 1000, ";") – $fp je otevřený soubor *.csv, 1000 je maximální počet znaků, který to načte a „;“ je tzv. oddělovač, podle kterého tato funkce rozhází jednotlivé řádky do polí.

Tato funkce pokaždé načte jeden řádek. Poté se pomocí SELECTu vyberou z databáze všechny prvky, které mají stejného výrobce, cenu odlišnou o 10 %, ale nenačtou se prvky, které mají původ ze stejného ceníku jako právě načtený produkt. Následně se se všemi vybranými prvky porovná název a pokud se s nějakým bude shodovat, tak v databázi zůstane produkt s nižší cenou a tento cyklus se opakuje pro každý prvek.

12. 4. 2. ed´ system Czech, a. s.

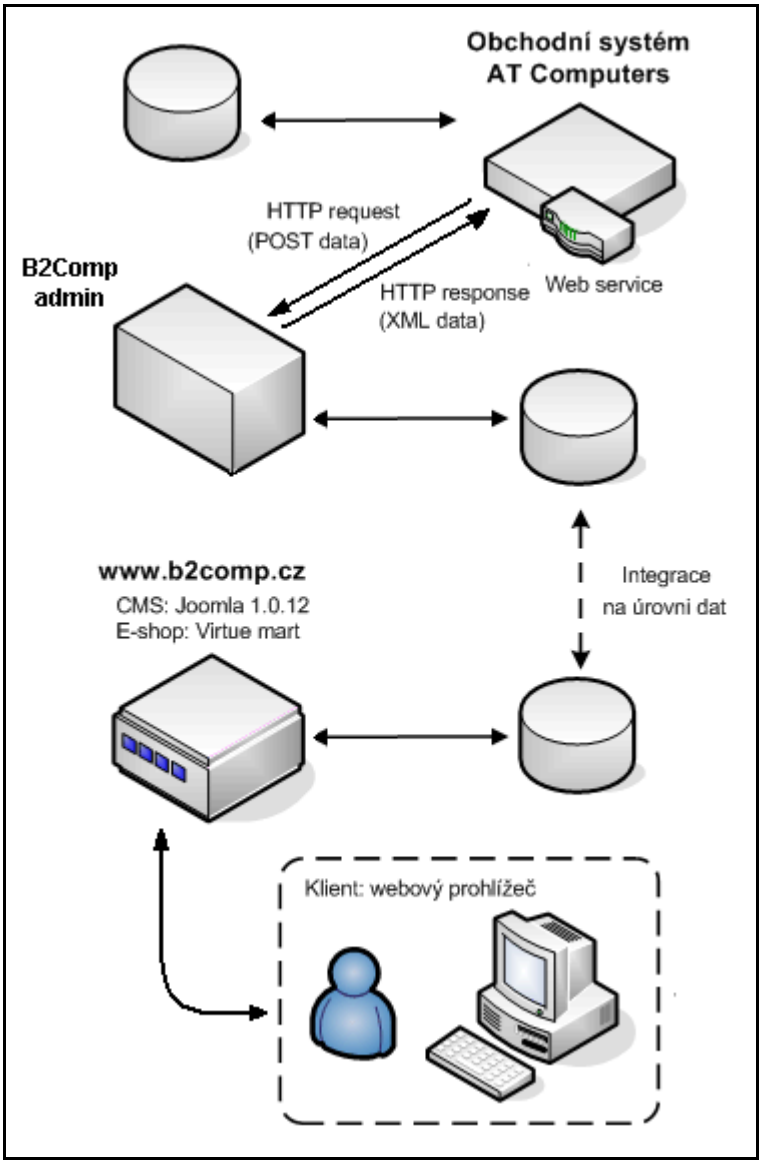
Společnost ed´ system Czech, a. s. poskytuje dealerům svůj ceník ve formátu *.mdb, což je databázový soubor programu Microsoft Access. Pro nahrávání používám knihovnu ADOdb, která je schopna pracovat s těmito databázemi: *MySQL, PostgreSQL, Interbase, Firebird, Informix, Oracle, MS SQL, Foxpro, Access, ADO, Sybase, FrontBase, DB2, SAP DB, SQLite, Netezza, LDAP,* and *generic ODBC, ODBTP*.

```php
function nahraj_tmp_databazi_edcz(){
    require("global_fce.php");
    spojeni();
    include('adodb/adodb.inc.php');
    include('adodb/adodb-pager.inc.php');    # load code common to ADOdb

    $conn = &ADONewConnection('access');     # create a connection

    $dsn = "Driver={Microsoft Access Driver (*.mdb)};
            Dbq=c:\\temp/cenikd2.mdb;Uid=root;Pwd=root;";
    $conn->PConnect($dsn);    # connect to MS-Access, northwind DSN
    $recordSet = &$conn->Execute('SELECT kod, zkr, nazev, dealer, zaruka FROM cenikd2');
    if (!$recordSet){
        print $conn->ErrorMsg();
    }
    else{
        while (!$recordSet->EOF){
            $a0= $recordSet->fields[0]; $a00= iconv("windows-1250", "UTF-8", "$a0");//kod
            $a1= $recordSet->fields[1]; $a11= iconv("windows-1250", "UTF-8", "$a1");//zkr
            $a2= $recordSet->fields[2]; $a22= iconv("windows-1250", "UTF-8", "$a2");//nazev
            $a3= $recordSet->fields[3]; $a33= iconv("windows-1250", "UTF-8", "$a3");//dealer
            $a4= $recordSet->fields[4]; $a44= iconv("windows-1250", "UTF-8", "$a4");//zaruka
            $table = mysql_query( "insert into `tmp_produkt` values
                ('$a00', '$a22', '$a33', '$a44', '$a11')");
            $recordSet->MoveNext();
        }
    }
    $recordSet->Close();
    $conn->Close();
    echo " Produkt: Dokoncen<br />";
```

Obrázek č. 13 – Ukázky zdrojových kódů 4

Na obrázku č. 13 je ukázána ukázka způsobu nahrávání databáze Access pomocí knihovny ADOdb. Jelikož je databáze rozmístěna do několika *.mdb souborů, bylo nejjednodušší řešení nahrát je všechny do dočasných tabulek v naší databázi, které budou existovat pouze po dobu nahrávání do trvalých tabulek a následně se opět smažou. Samotný algoritmus nahrávání produktů je založen na stejné bázi jako je to v předchozím případě s tím rozdílem, že se už pracuje s SQL dotazy.

12. 4. 3. AT Computers



Obrázek č. 14 – Schéma komunikace se společností ATComp

Na obrázku č. 14 vidíme způsob komunikace se společností AT Computers, od které ceník získáváme pomocí webových služeb.

Nejdříve jsem musel postupně zavolat webové služby na adresách:

http://www.atcomp.cz/webservices/ciselniky.asmx/Výrobci,

http://www.atcomp.cz/webservices/ciselniky.asmx/Podkategorie,

http://www.atcomp.cz/webservices/ciselniky.asmx/Kategorie,

http://www.atcomp.cz/webservices/zbozi.asmx/Cenik2.

K zasílání HTTP požadavku používám framework PEAR (PHP Extension and Application Repository) a z toho využívám knihovnu HTTP_Request. Použití této knihovny vidíme na obrázku č. 15.

```
// prilinkování knihovny funkcí HTTP_Request frameworku PEAR
include('HTTP/Request.php');

//vytvoreni HTTP dotazu na webovou službu spolecnosti ATComp
$req = &new HTTP_Request('http://www.atcomp.cz/webservices/
                          ciselniky.asmx/Kategorie');

//kategorie
$req->setMethod(HTTP_REQUEST_METHOD_POST);
$req->addPostData('strUzivatelskeJmeno', '');
$req->addPostData('strUzivatelskeHeslo', '');
$req->sendRequest();
$responsel = $req->getResponseBody();
$soubor=fopen("xml/kategorie.xml", "w");
fwrite($soubor, $responsel);
fclose($soubor);
$req->clearPostData();
```

Obrázek č. 15 – Ukázky zdrojových kódů 5

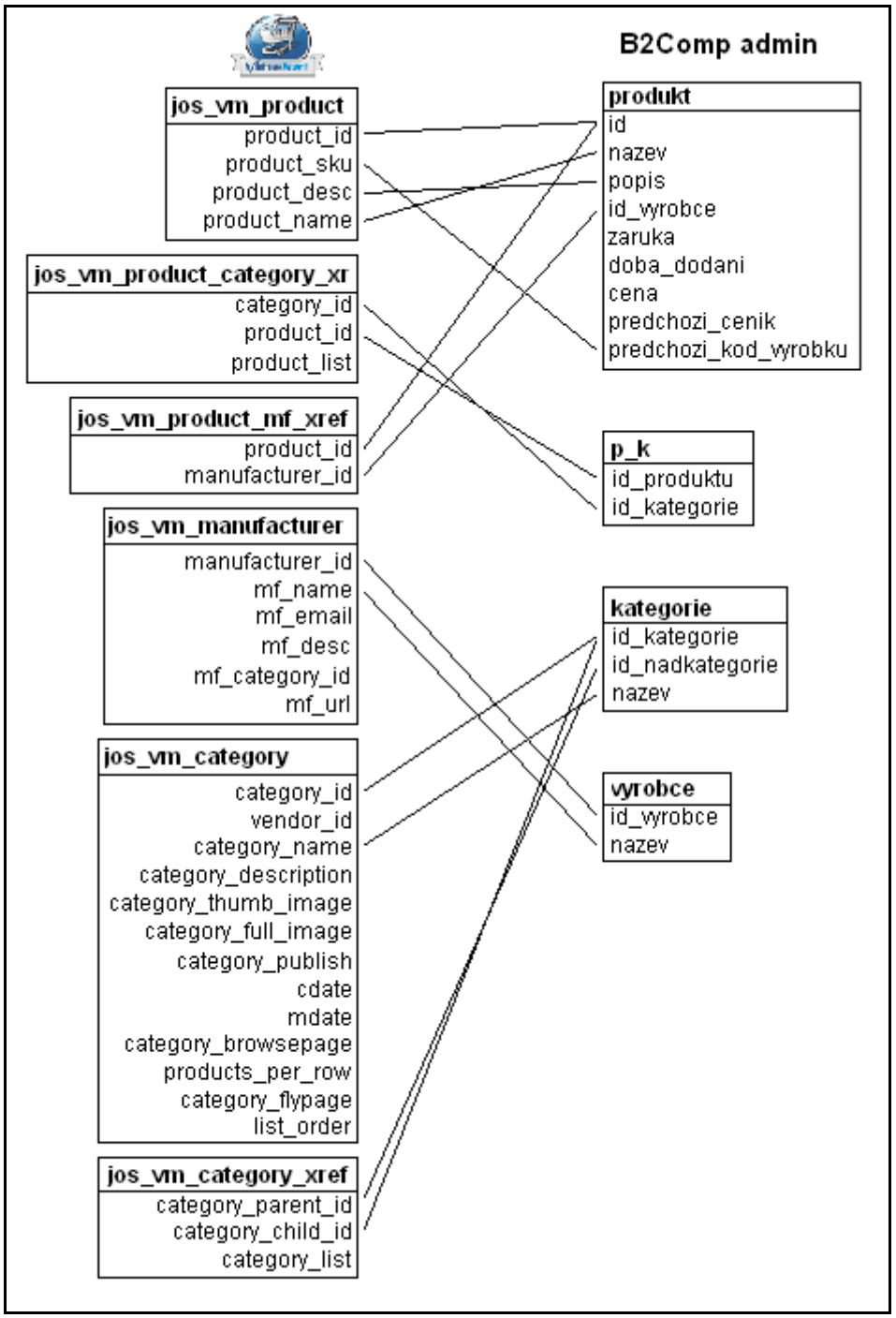
Na tyto dotazy mi webová služba vždy vrací XML dokument s požadovanými daty, které následně ukládám na pevný disk. K nahrání dat do mé databáze používám funkci *simplexml_load_file()* (použití v projektu je vidět na obrázku č. 16). Tato funkce je dostupná až od PHP verze 5. Samozřejmostí je, že algoritmus nahrávání se od předchozích případů nemění.

```
if (file_exists("xml/kategorie.xml"))
{
    $xml = simplexml_load_file("xml/kategorie.xml");

    foreach ($xml->xpath('//Table') as $item)
    {
        $kod = $item->kod;
        $nazev= $item->nazev;
        $table = Mysql_Query("INSERT INTO `kategorie` ( `id_kategorie` ,
            `id_nadkategorie` ,`nazev` ) VALUES ('$kod', 'root', '$nazev');");
    }
} else {
   exit("Nemohu nacíst soubor kategorie.xml");
}
```

Obrázek č. 16 – Ukázky zdrojových kódů 6

12. 5. Interface projektu

Jak bylo již řečeno, tak interface tohoto internetového obchodu bude zajišťovat kombinace CMR (redakčního systému) Joomla (http://www.joomla.org) a e-shopu VirtueMart (http://www.virtuemart.net). Na obrázku č. 13 vidíme, že mezi aplikací B2Comp admin a konečnou databází je pouze jedna vazba, která nám znovu zajišťuje perfektní flexibilitu v případě jakékoli změny (např. struktura databáze e-shopu VirtueMart). Nyní musíme provést převod dat z naší databáze do databázových tabulek VirtueMartu. Jelikož si můžeme ze stránek tohoto e-shopu stáhnout Developer manual (návod pro vývojáře), kde vidíme celou databázovou strukturu a vztahy mezi jednotlivými tabulkami, můžeme buď vyčíst, jak data uložit, nebo (což je jistější) postupným zkoušením zjišťovat, jakým způsobem, nebo lépe řečeno kam se data ukládají. Poté již jednoduchou konverzí dat pomocí SQL dotazů přeléváme data z naší databáze do VirtueMartu. Návaznost položek mezi tabulkami vidíme na obrázku č. 16.



Obrázek č. 17 – Návaznost mezi tabulkami

13. Závěr

Cílem této práce bylo zhodnotit současný stav a moderní trendy internetového obchodování. Mezi sebou byly srovnány nejvyužívanější skriptovací a programovací jazyky, databázové systémy a webové servery, přičemž jsem se u každého ze zmiňovaných přiklonil právě k tomu nejvyužívanějšímu z důvodu největší možné podpory ze strany uživatelů/programátorů. Zhruba bylo něco málo řečeno o bezpečnosti na internetu a nejběžnějších protokolech, jichž se k zajištění bezpečnosti využívá.

Praktickou částí této práce byla reorganizace struktury internetového obchodu společnosti B2Comp, kde v původním návrhu čerpali automaticky produkty do svého obchodu pouze od jedné společnosti a zbytek musel být doplňován ručně, což bylo dost pracné. V novém návrhu se chtělo čerpat ze tří ceníků najednou, přičemž do budoucna to nemusí být konečný počet, což znamená, že tento návrh musel řešit problém velké četnosti vazeb, což se nakonec podařilo.

V nynějším světe internetového obchodování jsou tři nejpoužívanější způsoby pro šíření ceníků – CSV („Comma Separated Values“), MDB (databáze Microsoft Access) a nakonec nahrávání pomocí webových služeb, které nám na dotazy poskytují soubory XML („*eXtensible Markup Language*“). Tato práce nastiňuje postup práce se všemi těmito formáty. Jsou zde ukázány funkce a postupy, které nám poskytuje PHP pomocí nichž můžeme tato data zkonvertovat do námi vybrané databáze (v této práci je ukázán převod do MySQL).

Největší problém se vyskytl při nahrávání různých ceníků, které obsahovali stejné zboží, protože stejný produkt v ceníku A se nemusí jmenovat úplně přesně jako v ceníku B, z čehož budou nejspíš vznikat ztráty, jež jsem se ale pokusil v co největší míře snížit výše zmiňovanými algoritmy. U plně automatického systému jako je tento se s minimálními ztrátami vždy musí počítat.

Zdroje

1. CASTAGNETTO, Jesus, aj. *Programujeme profesionálně PHP,* Brno: Computer Press 2004
2. Netcraft Ltd, May 2007 Web Server Survey [online], 2007, Dostupný z WWW: http://news.netcraft.com/archives/web_server_survey.html
3. The PHP Group, Usage Stats for April 2007 [online], Dostupný z www: http://cz2.php.net/usage.php
4. JUŘEK, Michael, Moderní integrace aplikací, Praha: Microsoft, s. r. o.

14. Seznam obrázků