Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačního inženýrství

Bakalářská práce

Jazyková lokalizace systému Joomla

Rastislav Šíša

! ! !

Místo této strany vložíte zadání bakalářské práce.
(Do jedné vazby originál a do druhé kopii)

! ! !

Čestné prohlášení

Prohlašuji, že svou bakalářskou práci "Jazyková lokalizace systému Joomla" jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu literatury na konci práce. Jako autor uvedené bakalářské práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne _____

Poděkování

Rád bych touto cestou poděkoval doc. Ing. Vojtěchu Merunkovi, Ph.D. za vedení mé práce, dále pak vlastní rodině, za trpělivost a podporu, a na závěr Aleši Haklovi a Ing.Michalu Stočesovi za odborné rady.

Jazyková lokalizace systému Joomla

National localization of system Joomla

Souhrn

Práce se zabývá lokalizací systému Joomla!. Internacionalizace a lokalizace programů je v současné době poměrně častým problémem, který je třeba řešit. Nejdříve je potřeba ale definovat co je vlastně takzvaný "CMS" (Content Management systém) – česky řečený "Systém pro správu obsahu". Je nutné do začátku uvést popis fungování a parametrů, které musí CMS splňovat. Kromě Joomla! zde ale existují i jiné neméně významné systémy pro správu obsahu, které v rámci open source tvoří nedílnou konkurenci tomuto systému, zde zmiňovaný například Wordpress a Drupal. Každý z těchto systémů má ve způsobu překladu svůj vlastní mechanismus, kterým ho lze překládat. Některý složitější, jiný jednodušší, jeden zaměřený na rychlost komunikace s programem, druhý na jednodušší překládání z pozice laika, tudíž je nutné je se systémem Joomla porovnat v tomto duchu. Následuje pak navržení aplikace, která by umožnila práci při překladu Joomla usnadnit. Pracovní název aplikace je v současné době JoomTee (Joom – Joomla, Tee – T jako Translator). Cílem aplikace je utvořit hromadnou databázi pro jazyky, která by umožnila generování konfiguračních souborů pro dané jazyky. Jejím přínosem pak je fakt, že přeložená slova, která se nyní v systému Joomla vyskytují, nebude nutné překládat v každém instalačním balíčku znovu a znovu.

Summary

The thesis is about localization of system Joomla!. Nowadays, iternationalization and localization is very often a problem and it is needed to be solved. At first we have to define, what the term CMS (Content management system) means. We have to observe which parameters and functions are defined for CMS. There are more of content management systems than Joomla, for example Wordpress and Drupal. Each of these systems has it's own mechanism, that allows the CMS to be translated and localized. Some ways are more difficult, some are easier. One way is more concerned about a speed of communication with the program, the other way may be easier for an amateur to translate it. So we need to compare and have an insight into some of the ways how other systems are localized. It is followed by designing an application, which could make translation of Joomla! a way easier. The working name of this program is JoomTee (Joom as Joomla, Tee as T means Translator). The main goal of this application is to create a common database, and generate configuration files which could be useful during a translation. Translated words, that are in a database, will not have to be translated in each package again, that is the main purpose for making this application.

Klíčová slova: Joomla, lokalizace, CMS, Wordpress, Drupal, internacionalizace, překlad, jazyk, JoomTee, Iniprocesor

Keywords: Joomla, localization, CMS, Wordpress, Drupal, internationalization, translation, language, JoomTee, Iniprocesor

• Obsah

1 Úvod

Problematika webových systémů pro práci s obsahem (takzvaných CMS systémů) je poměrně mladá, ale její úplné počátky můžeme datovat až téměř do roku 1989, kdy byl vydán produkt Lotus Notes verze 1.0. V té době splňoval pouze pár základních předpokladů plnohodnotného CMS systému. Dnešní systémy jsou daleko robustnější, pracují s daty dynamickým působem, využívají databází, může je ovládat jakýkoli i laický uživatel, který nemusí rozumět tomu, co se děje mezi jeho počítačem a serverem.

Jedním z těch nejznámějších a nejpoužívanějších CMS systémů dnešní doby je Joomla (v současné verzi 2.5.1. a stále se vyvíjí). Vydávána je jako opensource pod mezinárodní licencí GNU General Public Licence. To znamená, že kdokoli může Joomlu použít k vlastním účelům, má přístup k jejímu kódu. Co je open source a GNU General Public Licence bude vysvětleno později. Joomla! je využívaná odhadem v rámci tisíců webů, od komunitních webů po Její jazyková lokalizace je však poměrně krkolomná, není jiný způsob než ručně upravovat řádek po řádku v každém souboru zvlášť. Práce se tedy zaměří na popis CMS obecně, speciálně pak CMS Joomla!. Vedle Joomla! jsou tu i jiné známé redakční systémy, jejichž existenci se pokusí práce nastínit.

V rámci lokalizace systému Joomla! bylo nutné vytvořit pokusnou aplikaci, která by měla usnadnit práci překladatelům – vývojářům, kteří si budou moci pro vlastní jazyk vygenerovat konfigurační balíčky. Aplikace momentálně zachází s balíčky pro uživatelské rozhraní. To znamená, že je schopná lokalizovat uživatelské rozhraní a moduly k němu přiřazené. Administrátorské prostředí zůstává v původním překladu. Zdrojové kódy aplikace pak budou uveřejněny na www.github.org, což je poskytovatel služeb pro verzovací systém GIT (vyvinut původně Linusem Torvaldsem pro vývoj oparečního systému Linux).

2 Cíle a Metodika

Cílem práce je popsání způsobu lokalizace na nový jazyk, provedení dané lokalizace pro uživatelské rozhraní, administrátorské prostředí zůstává bez překladu, tedy v anglickém jazyce.

Metodika: Systém Joomla je třeba nainstalovat na vlastní server (v tomto případě se jedná o lokální počítač) a prozkoumat problematiku instalace. Následně vyzkoumat možnosti lokalizace modulů a jejího uživatelského rozhraní, porovnat s ostatními podobnými redakčními systémy. Ke studiu bude nutné použít existující instalační balíčky a dokumentaci daných redakčních systémů, aby bylo možné rozeznat způsoby, jakými jednotlivé CMS pracují se systém. V rámci toho je pak nutné porovnat způsoby lokalizace a nastínit vlastní způsob vytvoření lokalizace pro systém Joomla!. Vytvoření pokusné aplikace, která bude sama generovat konfigurační soubory, které se dají dále použít pro lokalizaci.

3 CMS

3.1 Historie CMS

(Kapitola je napsána podle vědomostí získaných ze zdroje [1]) World wide web (www) byl stvořen Timem Berners-Leem jako jednoduchý značkovací jazyk, který využíval internetu pro účely sdílení akademických listin a materiálů. Tuto užitečnou funkci plnil po několik let, kdy byla síť internet veřejnosti uzavřena a přístup do ní byl vyhrazen primárně pro členy akademické půdy. Když se síť internet otevřela veřejnosti v 90tých letech, nebyly na webové prezentace na svou dobu kladeny velké nároky.

První jednoduché stránky se skládaly převážně z textu a značky se používaly pro odstavce, hlavičky, občas tučné písmo nebo kurzivu. Byla zde i možnost přidání barevného pozadí a jeho určitých vlastností, například opakovatelnost a podobně. Na tehdejší standardy se dalo mluvit o výtečném výsledku.

Ve vývoji prohlížečů probíhal poměrně velký konkurenční boj, současně s tím byly firmy nuceny k vytvoření některých vyšších standardů pro značkovací jazyk XHTML. Dodnes různé prohlížeče počítají například odsazení jinak a podobně, takže dodnes se nepodařilo, aby se v prohlížečích zobrazovaly stránky naprosto stejně. A to i přes úsilí standardy sjednotit. Vývojem nového standardu tehdy velmi ztížilo a vzdálilo tvorbu webových prezentací amatérům, psát webové stránky byla tedy práce pro vývojáře. To vedlo k hledání jiných způsobů tvorby webových stránek, která by byla jednoduchá i pro absolutního laika.

Jedním z prvotních úspěšných pokusů o CMS byl Lotus Notes, který používal některé základní vlastnosti XHTML k formátování dokumentů v rámci svých ovládacích prvků. To byl pokrok oproti dřívějším způsobům tvorby. I když to mělo daleko do perfektního produktu dalo se říci, že to již obsahovalo některé základní vlastnosti Content Management Systému. Podstatnou roli zde hrálo oddělení dvou rolí – vývojáře a amatéra. Zatímco vývojář je schopen porozumět kódu, amatér to nepotřebuje. Jedním z cílů zde bylo tedy přiblížit tvorbu prostému uživateli, který nerozumí programování, ale orientuje se v obsahu. Dalším z hlavních cílů bylo pak vzetí v úvahu neustálé potřeby stránky aktualizovat a v jejich rámci organizovat velké množství neustále obnovovaného

materiálu.

S vývojem XHTML se musely paralelně vyvíjet jak servery, hardware na kterém programy generující XHTML, tak i samotné programy. Došlo tedy k vývoji skriptovacích jazyků, které svou činnost prováděly právě na straně serveru(a ne klienta). Tyto jazyky se zakládaly na konceptech programovacích jazyků vycházejících z jazyka C, ale do nich byly zahrnuty vlastnosti, které byly určeny ke generování XHTML. Jedním z nejznatelnějších vývojových kroků bylo vytvoření skriptovacího jazyka PHP. Jak se vyvíjely, získaly skriptovací jazyky některé prvky, které jsou specifické pouze pro prostředí Webu.

Dalším milníkem bylo vyvinutí komplexních systémů, které byly navrženy speciálně pro organizaci materiálu a jeho jednoduchou a „hladkou“ prezentaci. Vývojem otevřených Open Source systémů pak bylo umožněno lidem s malým rozpočtem vytvářet vlastní stránky na profesionální úrovni. Mohli vytvářet stránky, které mohly vypadat k světu a prezentovat jejich firmu. Přesto zpočátku bylo těžké najít systém, který by splňoval podmínky pro co nejjednodušší tvorbu stránek.

Prvním průlomovým CMS bylo Mambo 4.5. Dalo se nainstalovat během pár minut a měli jste framework pro kompletní webovou stránku s navigací a některými dalšími užitečnými prvky. Mambo poskytovalo vlastní šablony, což umožňovalo, že i prostý text vypadal dobře. Když si člověk trochu připlatil, bylo možné nechat si vyrobit šablonu, která vypadala profesionálně a přesně podle představ zadavatele. Nebylo těžké zobrazovat články jakéhokoli typu. Mambo mělo některé základní dovednosti k vytváření a publikování článků. Jeho pozdějším nástupcem a zároveň odnoží je známý CMS systém Joomla, která má ve světě CMS dobré jméno.

3.2 Zásadní prvky CMS

(Kapitola je napsána podle vědomostí získaných ze zdroje [1]) Nyní, když jsme definovali CMS jako systém pro správu obsahu na webu, je na čase upřesnit další jeho možnosti. Mambo například bylo ve své době příliš orientováno na práci s dokumenty, to bylo i jeho velkým a očividným omezením. Zatímco každá stránka obsahuje nějaký text, pouze některé jsou tím omezeny. I tam, kde je text prioritní záležitostí, starší systémy jsou

zatlačeny na pokraj svých možností poptávkou po větší flexibilitě v omezení toho, kdo má přístup k jakému materiálu a kdo může nebo nemůže dělat některé věci v rámci systému.

Zatímco Mambo samo o sobě mohlo být nainstalováno a používáno s velkou řadou funkcí, jeho úspěch ve světě CMS přinesla jeho možnost rozšíření o různé moduly a funkce. Bezpočet těchto funkcí a modul byl vyvíjen mimo hlavní vývoj Mamba. Existence této půdy pro rozvoj přídavných možností byla velmi oceňována mnoha uživateli Mamba. Moduly byly vytvářeny pro běžné požadavky, řešení běžných problémů, v rámci webových funkcí a byly dostupné k instalaci do systému Mambo. Pro neobvyklé případy, kterým nestačilo obecné existující řešení, se daly moduly poupravit a nebo napsat nové v rámci Mambo Frameworku. Velkou výhodou pak byla možnost navrhování rozsáhlejších šablon pro každý prvek systému a jeho zařazení pak v rámci stránky do celkového designu či jiných jejích služeb.

Využití CMS se rozrostlo současně s tím, jak se začal rozvíjet svět Webu a jeho interaktivnost. Stránky například jako Amazon, E-bay, různé e-shopy, inspirovaly tvůrce CMS systémů a poskytly jim nápady k inovacím v systémech, které se netýkají pouze textu a jde tedy například o to prodat nebo koupit zboží.

Základní vlastnosti, které jsou tedy od CMS v rámci požadavků uživatelů očekávány:

1. **Kontinuita a interaktivnost** – v rámci webových aplikací musí uživatel mnohdy pokračovat skrze více kroků k získání výsledku procesu, jehož je v tu chvíli účastníkem. Kroky v rámci například v rámci internetového bankovnictví musí být zabezpečeny tak, aby se mohl uživatel dostat ke svému účtu, provádět transakce a měnit nastavení účtu. V žádném případě nesmí být v podobném systému mezera, která by umožnila zneužití jeho citlivých dat.
2. **Správu uživatelů** – v každém systému, dříve či později, musíme definovat kdo je jeho uživatelem, mnohdy nechceme, aby náš systém navštívil někdo nepovolaný. CMS tedy poskytuje přihlašovací formulář, který ověří identitu daného člověka. Snahou je pochopitelně omezit množství kódu dané funkce, ale aby zároveň fungovala pro velký počet uživatelů.

3. **Správa přístupu** – přidělení práv (*Role-Based Access Control*) jednotlivým uživatelům nám umožňuje kontrolovat kdo a co nám v našem systému může dělat. Každý uživatel má v rámci systému svou roli. To nám umožňuje přidělit jiná práva pro administrátora, jiná pro obyčejného uživatele či osobu, která stránky jen navštíví jako host.
4. **Správa rozšíření** – redakční systém se stává užitečným ve chvíli, kdy jej můžeme podle vlastních potřebo rozšiřovat. V dnešní době v systémech jako je například Joomla nebo Wordpress či Drupal, máme možnost si vytvořit webovou prezentaci i jako absolutní laik. Proto je nutná jednoduchá instalace přídavných modulů, které náš systém učiní pro nás použitelný.
5. **Zabezpečení a ošetření chyb –** běžný uživatel nemůže sledovat zabezpečení systému, pokud se systémem pracuje, čeká od něj zabezpečenost s naprostou samozřejmostí. Proto jsou v CMS instalovány moduly, které například usnadňují práci administrátorů a chrání stránky od napadení roboty šířícími spam (běžný problém). Dále pak musí CMS vědět jak reagovat na chyby.

Kromě těchto kritických vlastností CMS systémů jsou ale pro uživatele směrodatné i jiné vlastnosti, bez kterých by se v běžné praxi systém neobešel. Zvláště pokud chceme systém dále jednoduše rozšiřovat. Aby toto bylo splněno, jsou zapotřebí následující kriteria:

1. **Efektivní a pohodlné ovládání kódu** – Framework by se měl skládat z několika zdrojových souborů. Je jasné, že ty se pak volají pouze ve chvíli, kdy jsou potřebné, systém musí zároveň umět nakládat s extra modulovými soubory a rozšířeními.
2. **Databázové rozhraní** – Mnoho webových aplikací potřebuje přístup k databázi ať tím či oním způsobem. Přístup k databázi musí fungovat efektivně. Systém jako takový potřebuje pro své fungování databázi. Zatímco PHP přístup poskytuje k různým databázím, je zde mnoho možností, které CMS framework splňuje, aby naplnil běžné potřeby v rámci práce s daty. To platí jak pro rozšíření a pluginy tak i pro framework samotný.

3. **Cache** – Cache jsou používány v mnoha různých kontextech v rámci internetových procesů. Nejproduktivnějšími oblastmi v rámci cache jsou objektové a XHTML cache. Rychlost operací a proces načítání benefitují z dobře implementovaných cache. Pro CMS framework to znamená, že musí zajistit cacheovací mechanismy, které budou „lehké“ a jednoduché pro použití.
4. **Menu** -menu jsou běžnou součástí webových stránek, speciálně když se v širším slova smyslu tvoří navigační lišty, které jsou ve své podstatě jen seznamem odkazů. Není smyslem Frameworku, aby obsahoval čisté XHTML soubory v pravém slova smyslu, protože musí brát v ohled různé šablony a rozšíření. Naopak je vyžadováno, aby měl framework šablonovací systém, který se dynamicky přizpůsobuje přijatým informacím a ty pak formuje do XHTML souborů podle šablon. Stejně tak podle šablony dynamicky formuje i jednotlivá menu.
5. **Jazyky** – V poslední době , aspoň tedy minimálně, by se měla brát při vývoji v potaz potřeba překládat software do jiných jazyků. Včetně těch, které obsahují vícebajtové znaky. Nyní je v široké vývojářské komunitě odsouhlaseno, že součástí řešení je používání mezinárodního standardu UTF-8. Je určitě nutné, aby měl systém vlastní mechanismus, který překlad textu umožní. S otázkou překladu ale vyvstalo více problémů daných požadavky na jazykovou podporu, která musí odpovídat přesným pravidlům internacionalizace a lokalizace. Internacionalizace znamená vestavět do systému možnosti podpory různých změn v rámci CMS, jejichž nejdůležitějším prvkem je právě změna jazyka. Lokalizace je speciální rozvinutí specifických charakteristik pro danou lokalitu (území) , které byly internacionalizovány, do CMS systému. Kromě samotného jazyka a podobně jsou to pak formáty data, peněžní měny, čísel a dalších.

3.3 Open Source

S nejznámějšími a nejpoužívanějšími CMS systémy po celém světě jako je právě Joomla!,Wordpress nebo Drupal je spjat pojem Open source. Je ale třeba upřesnit, co Open source znamená.

3.3.1 Co je Open source?

(Kapitola je napsána podle vědomostí získaných ze zdroje [2]) Open source, jako termín, neznamená pouze otevřený přístup ke kódu. Distribuce programů pod opensource musí splňovat následující kritéria:

- **Volná redistribuce**

Autor a tedy majitel licence musí mít právo svůj software volně šířit, dělat jeho kopie a distribuovat ho ať už za komerčním či nekomerčním účelem. Samozřejmým právem musí být i redistribuce jednotlivých částí vlastního software jako součástí jiného software či paralelně s jiným software. Vlastník licence má oprávnění poskytovat podlicence ať už zištně či nezištně, ale poskytovatel licence nemá právo podmiňovat si zařízení podlicence vlastníkem peněžitou odměnou. *„Pouze další distribuce počítačového programu, za jehož poskytnutí byla již odměna uhrazena, nesmí být ze strany původního poskytovatele zpoplatňována.“*[2]

- **Zdrojový kód**

Zdrojové kódy software musí být poskytnuty nabyvateli licence. Tomu poté musí být právně umožněna redistribuce programu jak jako spustitelný soubor, tak i jako zdrojový kód. Pokud není zdrojový kód nabyvateli licence poskytnut přímo, musí být informován o zdroji, ze kterého se dá tento zdrojový kód získat bez nákladů, které by byly zbytečné. Zdrojový kód poskytnutý nabyvateli licence je nutno předat ve formě dále upravitelné tak, aby s ním mohl nabyvatel licence nakládat podle svých záměrů a těm ho přizpůsobit. Zdrojový kód musí být navíc předáván ve srozumitelné podobě, není dovoleno jakkoli stěžovat práci s ním, či ho poskytovat v různých etapách překladu.

- **Odvozená díla**

Zdrojový kód musí být licenčně ošetřen tak, aby ho mohl nabyvatel licence měnit a upravovat. Tento upravený software pak musí být dále distribuovatelný v rámci stejné licence jako byl program původní.

- **Integrita zdrojového kódu autora**

Pokud je možné distribuovat s původní verzí software i opravné balíčky (patch), může být kód omezován licenčními podmínkami. Tyto opravné balíčky lze pak použít při překladu(kompilaci). Distribuce software kompilovaného z modifikovaného zdrojového kódu původního software musí být výslovně povolena. Licenční podmínky mohou obsahovat omezení, které určí, že modifikovaný software by se měl vydávat pod jiným jménem či jinou verzí daného programu.

- **Zákaz diskriminace osob či skupin**

V licenčních podmínkách se nesmí nacházet omezení, které by omezovalo jakoukoli skupinu lidí či konkrétní osoby, licenční podmínky nesmí nikoho diskriminovat.

- **Zákaz diskriminace podle oboru činnosti**

Pravidla, která by znamenala omezení užívání daného software pouze pro nějaký určitý obor činnosti jsou v rámci licenčních podmínek nepřípustná. Není tedy možné, aby se program v rámci open source mohl používat například pouze ve školství, nebo pouze pro výzkum molekulární biologie a podobně.

- **Aplikovatelnost licenčního ujednání**

Licence musí být postavena tak, aby pokryla různé možnosti v rámci pracovních vztahů, které by se daného software týkaly. Jejich obecnost musí být pak aplikovatelná tak, aby nebylo nutné podmínky rozšiřovat o dodatky či jiné speciální licenční smlouvy, které by původní licenci rozšiřovaly.

- **Licenční podmínky nesmí být určeny pouze pro konkrétní softwarový produkt**

V licenčních podmínkách daného software nesmí být odlišnosti vázané na vydání programu v závislosti na distribuci s jinými programy. K jednomu danému programu musí mít všichni nabyvatelé licence shodné licenční podmínky (omezení či výhody z nich plynoucí), ať už je software vydán jako samostatný a nebo současně v distribuci s jiným software.

- **Licenční podmínky nesmí zasahovat do právních vztahů k jinému software**

Licenční podmínky nesmí žádným způsobem zasahovat do práv a povinností k jiným počítačových programům, jež jsou distribuovány společně s předmětným počítačovým programem, na něž se licenční podmínky vztahují. Např. určité licenční podmínky nesmí stanovit, že všechny počítačové programy distribuované na jednom hmotném mediu se musí řídit licenčními podmínkami splňujícími kriteria pro Open Source software. Podmínky obsažené v licenci nesmí v žádném případě ovlivňovat práva a povinnosti jiných software, které by byly v distribuci současně s daným software, který je podmínkami vymezen. To znamená například, pokud bychom měli paměťové médium obsahující více programů, které se pro tuto chvíli distribuují dohromady s daným software, nemůžeme licenčními podmínkami vymezit obsah celého média jen jako open source. Licence se nevztahuje na ostatní software na médiu.

- **Technologická neutralita licenčních podmínek**

Licenční podmínky nesmí svým ustanovením určit technologii, na které by byl daný software závislý.

3.3.1.1 General public license

Přeloženo z angličtiny z oficiálních stránek [3]:

Nikdo by neměl být omezen softwarem, který používá. Jsou čtyři pravidla „svobody“, která musí být stejná pro každého uživatele:

- možnost a volnost používat software k jakémukoli účelu
- možnost volně upravovat software k vlastním potřebám
- možnost volně sdílet software se svými bližními, sousedy a dalšími
- možnost volně sdílet modifikovaný software

Licence GPL se dále vyvíjí tak, aby ochránila uživatele i autory v rámci volného používání software vydaného pod touto licencí.

3.4 Lokalizace

„Pojmem internacionalizace se rozumí proces úpravy existujícího programu tak, aby byl schopen pracovat v (jazykovém a kulturním) prostředí uživatele, respektovat místní zvyklosti ohledně reprezentace času, čísel apod., a v ideálním případě s uživatelem komunikovat v jeho jazyce.

Přitom se klade důraz na to, aby byl program na konkrétním prostředí nezávislý a bylo jej tedy možné používat i v jazykových prostředích, pro která nebyl původně určen, bez zásahu do programu samotného. Úprava takto připraveného programu pro konkrétní prostředí (což znamená především překlad textů, které program používá), se nazývá lokalizace.“[4]

3.5 Joomla

3.5.1 Představení Joomla

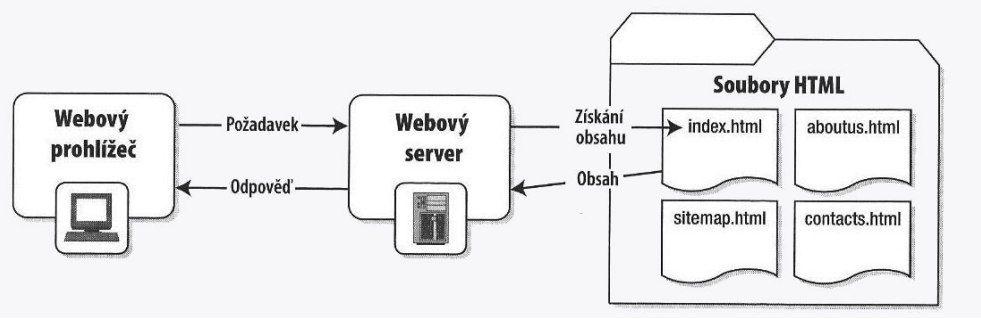
Joomla patří mezi tři nejpopulárnější Open Source redakční systémy (redakční systém = CMS). Je vydávána pod licencí GNU GPL, tedy General Public License. Systém je určen jak pro jednotlivce, tak i pro střední i velké firmy. V rámci Joomla! je možno vytvářet velmi snadným způsobem webové stránky, od jednoduchých blogů po rozsáhlé e-shop portály, internetové časopisy a podobně.[5]

„Jedná se o původně komerční produkt, který byl později vydán pod svobodnou licencí.“ [6]

3.5.1.1 Způsob fungování

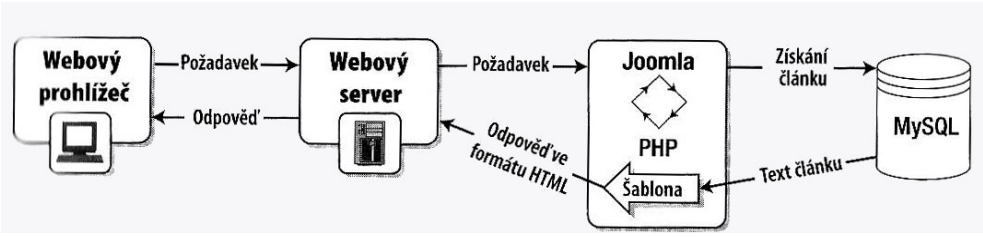
(Kapitola je napsána podle vědomostí získaných ze zdroje [7]) Pokud porovnáme s běžnými statickými weby (v HTML), CMS systém funguje daleko komplikovanějším způsobem, přesto není těžké ho používat, ale je nutno porozumět některým základním principům, které fungují v rámci tvorby a fungování webových stránek. Poté by měl být uživatel schopen nakonfigurovat si publikační systém přesně podle toho, jak potřebuje, aby se jeho příspěvky zobrazovaly.

V základním modelu fungování webového serveru **(viz. Obrázek 3.4.4.1.a. klasický server)** můžeme vidět, že z webového prohlížeče jde uživatelův požadavek směrem k serveru, ten požadavek vyhodnotí a z úložiště pak vybere statický XHTML soubor, který obsahuje požadované informace. Poté, co zjistí, zda vyhovující statický soubor existuje, přeposílá jej server směrem ke klientskému počítači v nezměněné podobě, klientský počítač pak informace zobrazí podle prohlížeče, který používá.



Obrázek 3.4.4.1.a. Statický web[7]

Pokud se podíváme na události zobrazené na obrázku 3.4.4.1.b., můžeme vyvodit jaký je rozdíl mezi fungováním CMS systému Joomla a statickým webem jako takovým. V adresním řádku by teoreticky uživatel neměl poznat rozdíl, uživatel žádá obsah a není na něm, aby sledoval celý postup svého požadavku. Poté co požadavek zadá skrze prohlížeč webovému serveru, webový server odešle data programu psanému ve skriptovacím jazyku PHP, v tomto případě je tím programem právě Joomla!, která pak požadavek zpracuje. CMS se tedy dotáže MySQL databáze a získá kýžený obsah. Po zpracování dat CMS systém generuje pro webový server XHTML soubory, které jsou vytvářeny podle šablon. Celý systém zpracovává data z databáze z textové podoby, stačí tedy změnit pouze šablonu, podle které se data zobrazí, a je možno naprosto změnit vzhled generovaného souboru a tedy stránky zobrazené na straně klienta. XHTML soubory jsou pak obratem přeposílány na klientský počítač, který zpracuje získaný soubor v prohlížeči. Klient v tuto chvíli ví akorát o tom, že vyslal požadavek a obdržel odpověď, celý zbytek procesu je mu skryt.



Obrázek 3.4.4.1.b. - Fungování CMS Joomla[7]

3.5.2 Historie

(Kapitola je napsána podle vědomostí získaných ze zdroje [5])Podle českého portálu Joomlaportal.cz je první verze Joomla časově zařazena do roku 2005, kdy ve verzi 1.0 spatřila světlo světa. Kořeny tohoto CMS ale sahají daleko dále, a to ke staršímu CMS zvanému Mambo z roku 2000. To vznikalo pod záštitou společnosti Miro Corporation. Původně bylo Mambo zamýšleno jako takzvaný „closed source project“, komerční CMS systém, to ale neslavilo valného úspěchu, vývoj stagnoval a tak bylo nuceni projekt otevřít jako open-source (v dubnu 2001). Open-source zaručuje otevřenost kódu pro kohokoli. Po zveřejnění kódu tohoto velmi pokročilého CMS systému netrvalo dlouho a na trhu se objevila i česká lokalizace (2002 s verzí 4.0).

Joomlaportal.cz pak dále zmiňuje, že systému Mambo se jeho rozšířeností a oblíbeností mezi lidmi po celém světě a webu, povedlo zavést nový průmysl. *„Systém Mambo si získal popularitu po celém světě a dal vznik celému průmyslu dodavatelů poskytujících rozšíření a šablony. Vývojářská komunita v prostředí, kde lidé mohli svobodně šířit své nápady a zdrojové kódy vzkvétala.“*[5]

V této době měl systém Mambo obrovskou oblíbenost, dalo se málem mluvit o tom, že byl nejoblíběnějším Opensource CMS celosvětově. Recenzenti o něm hovořili téměř vždy kladně.

Zlomovým je rok 2005, kdy se vývojáři Mambo ze sféry open source nepohodnou s neziskovou organizací, která dělá vývoji tohoto systému dozor. Výsledkem neshody je nevyhnutelné rozdělení a část programátorů se vrhá do vývoje odnože Mambo systému. Zanedlouho vzniká tedy první verze CMS Joomla (1.0), která je v té době s Mambo CMS kompatibilní téměř na 100% alespoň v rámci jádra. Nový CMS má modernější správu obsahu, přívětivější uživatelské rozhraní.[5]

„Joomla nebyla vytvořena úplně na zelené louce. Je to odnož předchůdce nazvaného Mambo. Kvůli sporům se vývojový tým rozštěpil na dva tábory, jeden zůstal a vytváří Mambo dál, druhá část má svou Joomlu. V určité fázi byly oba systémy volně zaměnitelné, v současné době je kompatibilita na úrovni asi 60% a s přibývajícím časem

bude klesat. “[6]

Systém Mambo si díky potížemi s organizací vysloužil částečný úpadek v popularitě, Joomla ho zastínila. Ačkoliv je Mambo stále vyvíjeno, Joomla s ním svou podobnost ztrácí.[5]

3.6 Drupal

3.6.1 Historie a popis produktu

(Volný překlad z oficiálních stránek)[8]

V roce 2000 bylo trvalé připojení k internetu vzácností mezi studenty Antverpské univerzity, takže Dries Buytaert a Hans Snijder sestavili bezdrátové přemostění mezi studentskými kolejemi, aby nasdíleli Hansovo ADSL připojení mezi osm studentů. V této době to byla situace jako tato v podstatě otázkou luxusu, ale něco chybělo: Nebyl zde žádný prostředek ke sdílení maličkostí a k vedení diskuse.

To inspirovalo Driese k sestrojení malé webové stránky s vestavěnou nástěnkou, která dovolovala skupince přátel zanechávat si navzájem vzkazy o stavu sítě, o tom, kdy půjdou na večeři, a nebo o novinkách ve světě, které stály za zmínku.

Software neměl jméno, dokud se Dries po dokončení studia univerzity neodstěhoval. Skupinka přátel se tedy rozhodla stránku uveřejnit a dát online, aby mohli zůstat v kontaktu, sdílet důležité objevy, a vyprávět si historky z osobního života. Zatímco hledali vhodné jméno pro webovou doménu, Dries ustanovil „drop.org“ poté, co se přepsal, když zkoušel, jestli je volná „dorp.org“. Dorp je dánský výraz pro vesnici, což bylo schváleno jako odpovídající název pro jejich malou komunitu.

Jakmile byl drop.org spuštěn na Webu, jeho publikum se změnilo s tím, jak členové začali hovořit o nových webových technologiích jako moderování, publikování, hodnocení a rozdělení autentizace. Drop.org se změnil na osobní pokusné prostředí řízené diskusemi a tokem myšlenek. Diskuse o těchto webových technologiích byly rovnou zkoušeny na drop.org jako nové přídavné funkce k běžícímu software stránky.

O rok později, v lednu roku 2001, se Dries rozhodl vypustit do světa software stránky drop.org pod jménem „Drupal“. Účelem bylo umožnění ostatním tento software používat a jako experimentální platformu ho rozšířit mezi více lidí, kteří by mohli rozšiřovat jeho funkce a pomoci ho vyvíjet. Jméno „Drupal“, které se tak i vyslovuje, je odvozeno z anglické výslovnosti Dánského slova „druppel“, které znamená „kapka“.[8]

„Drupal je open source redakční systém, tedy volně dostupný software, který staví na několika základech, které jsou důležité pro jeho fungování a vývoj:

- *Modularita – Chcete blog? E-shop? Fórum? Korporátní web? To vše Drupal umožňuje díky svému modulárnímu systému: Malé, ale stabilní a rychlé jádro s dobrým rozhraním a moduly, na kterých staví. Každý může vytvořit vlastní modul, seznam modulů je udržován na domovské stránce Drupalu.*
- *Kvalita – Do jádra Drupalu se nedostávají neověřené patche, jádro má rovněž velmi dobře navrženou strukturu. To z něj dělá bezpečný a stabilní systém*
- *Open Source – GNU/GPL license, PHP programovací jazyk, podpora pro MySQL a PostgreSQL, připravovaná podpora pro MS SQL a Oracle.“*[9]

3.7 Wordpress

3.7.1 Historie a popis produktu

(Kapitola je napsána podle vědomostí získaných ze zdroje [10]) WordPress je CMS systém, který byl naprogramován v roce 2003. Jeho začátky byly založeny jen na sporém kódu a používalo ho pouze omezené množství lidí z komunity, která byla zasvěcena do projektu. Postupem doby se jeho vývoj a oblíbenost zvýšily a projekt prodělal velké množství změn. V současné době je jedním z nejpoužívanějších CMS na světě v rámci takzvaných “blogů“. Používá jej až několik stovek tisíc autorů obsahu blogů a čtenáři, kteří na podobné stránky přicházejí se dají počítat v milionech.

K jeho vzniku vedl nápad stvořit elegantní, osobní CMS systém, s dobrou strukturou. Prostředky k sestavení tohoto systému byly pak PHP a MySQL a licencován je pod General Public Licence. Vznikl z původního systému b2/cafelog. Wordpress je systémem s poměrně krátkou historií, úplné počátky můžeme však najít až v roce 2001. Jeho produkce se je zaměřena na webové standardy a stabilitu tak, aby co nejvíce vyšel vstříc uživatelům.

„Je to velmi vyspělý a stabilní produkt zaměřující se na uživatele a webové standardy čímž se odlišuje od jiným nevyspělých publikačních systémů.“[10]

Zlomovým rokem v rámci historie verzí je pro Wordpres rok 2005, kdy vydáním verze 1.5 přilákal více než 900 000 uživatelů internetu, kteří si tuto verzi stáhli. Současně s tím, pak odstartovala i hostingová služba wordpress.com, která je nyní k dispozici pro kohokoli, kdo si chce vytvořit osobní blog a nemusí cokoli instalovat či konfigurovat v rámci serveru, ale software je již pro uživatele připraven jako kompletní služba. Člověk tak může téměř okamžitě začít s psaním vlastních článků, s šablonováním systému a prací se službou obecně.[10]

4 Lokalizace Joomla!

4.1 Instalace Joomla

4.1.1 Instalace Joomla! CMS na lokální počítač

4.1.1.1 Příprava pro instalaci:

Pro instalaci systému Joomla! (verze 2.5.1) jsem vybral nástrojů, které svým nastavením vyhovují nejlépe pro vývoj na vlastním počítači. Samotný CMS Joomla! můžeme stáhnout na stránkách www.joomla.org. Nástroj, který jsem vybral jako vývojové prostředí na vlastním počítači, nalezneme na http://www.apachefriends.org/en/xampp.html , funguje jak pod systémem Linux, Windows, tak i pod MacOS či Oracle Solaris. Má tedy široké možnosti v rámci kompatibility na všech platformách. Obsahem tohoto balíčku je PHP (skriptovací jazyk), MySQL databáze a Apache server.

Jeho instalace je velmi jednoduchá a balíček si v rámci systému nastaví všechny potřebné parametry. Na Linuxu (pro návod je využíváno operačního linuxového systému odvozenného z Debianu a to Ubuntu 11.10) je potřeba pak přidělit speciální práva složce, kam se budou nahrávat soubory pro jakýkoli testovací web. V našem případě je to složka */opt/lampp/htdocs*. Přidělíme jí tedy práva jako superuživatel v terminálu *sudo chmod 777 /opt/lampp/htdocs* a vytvoříme v ní složku *mkdir /opt/lampp/joomla* joomla (*mkdir /opt/lampp/joomla)*. Té přidělíme stejná práva, která musíme aplikovat i na podřízené soubory (tento krok by na veřejné síti nebyl bezpečný, protože chmod 777 umožňuje čtení, zapisování a spouštění souborů pro jakéhokoli uživatele, tedy i uživatele, který tato práva mít v ideálním případě nemá, pokud není vývojář). Pro vývoj na localhost je to nejpohodlnější řešení, není pak tedy třeba řešit práva jednotlivých souborů při instalaci. Do složky „joomla“ pak tedy zkopírujeme soubory z komprimovaného archivu, který jsme stáhli ze stránek Joomla.

Před otevřením prohlížeče je nejdříve nutné spustit samotný Xampp server s MySQL, PHP a Apache. Do terminálu tedy jako superuživatel vložíme příkaz *sudo /opt/lampp/lampp start* a pokud nenastane žádná komplikace a systém nenahlásí chybu, měl by se na výstup terminálu vypsat tyto řádky:

„Starting XAMPP for Linux 1.7.7...

XAMPP: Starting Apache with SSL (and PHP5)...

XAMPP: Starting MySQL...

XAMPP: Starting ProFTPD...

XAMPP for Linux started. “

Tím je spuštěn server a připraven pro testování a vývoj webových aplikací.

Dalším krokem pro instalaci je vytvoření databáze pro naši aplikaci. Joomla, podobně jako Wordpress nebo Drupal, si vytváří tabulky sama pomocí skriptu. Jediné, co budeme potřebovat, je vytvoření databáze a přidělení uživatelského účtu a hesla. V okně prohlížeče zadáme do adresního řádku adresu *http://localhost/phpmyadmin* a spustíme tím PhpMyAdmin. PhpMyAdmin je jeden z nástrojů pro správu MySQL databáze a je také součástí instalace xampp serverového balíčku. V poli pro zadání jména nové databáze vložíme jméno (v tomto případě například joomla) a vytvoříme databázi. Znaková sada není třeba volit, Joomla si ji poté zvolí sama při tvorbě tabulek. Nyní je vše připraveno pro samotnou instalaci.

4.1.1.2 Instalace:

V okně prohlížeče zadáme do adresního řádku *http://localhost/joomla* (za lomítkem je název složky, ve které se v htdocs nachází soubory CMS Joomla!). Samotná instalace je velmi intuitivní a „userfriendly“. Jako první se nám nabízí obrazovka s volbou jazyka, zde si můžeme již předem vybrat z mnoha základních jazyků, pro které je Joomla! lokalizována. Pokud se místo první obrazovky vypíší pouze chybové řádky, prvním důvodem, který bychom měli ověřit jsou nastavená práva souborů v místě, kde jsou uloženy. Tlačítkem následující přejdeme na kontrolu nastavení našeho vývojového prostředí na lokálním serveru (localhost). Zde jsou zobrazena dvě oddělení, která souvisí s budoucím během aplikace. V prvním rámečku jsou položky, které jsou nezbytně nutné pro běh, v druhém pak máme položky, které sice pro běh nejsou vyžadovány doslovně, ale jejich nastavení souvisí s budoucími mírnými odchylkami chování, ty ale nebudou mít závažný vliv na chod. Povšimněme si například INI parseru (funkce, která využívá v php funkce parse_ini_file($file)), tento se pak používá pro zpracovávání .ini souborů, které

fungují jako konfigurační. V INI souborech tedy pak například ukládáme data pro jazykovou lokalizaci. Dalším důležitou položkou je také configuration.php, soubor, který musí být v době instalace zapisovatelný (opět pozor na nastavení práv), zde se poté ukládají informace o připojení k serveru a databázi.

V méně důležitých položkách pak budeme mít zpravidla červeně vyznačeny jen položky jako je zapnutý chybový výstup, který pochopitelně pro vývoj na straně lokálního počítače potřebovat můžeme.

Pokud nenastal žádný problém, přesuneme se opět na následující obrazovku. Ta obsahuje znění General Public License, kterou si můžeme přečíst v plném znění. Po přečtení se posouváme dál k nastavení databáze.

Zde si vybereme typ databáze, kterou používáme, v našem případě volíme MySQL. Hostitelským počítačem je zpravidla localhost (tedy lokální počítač). Již při nastavování databáze v aplikaci PhpMyAdmin jsme nevytvářeli žádného nového uživatele, tedy uživatelem bude root. Heslo pro tohoto uživatele je v balíčku XAMPP standardně prázdné (pro jednoduchost). Do pole databáze zadáme název naší databáze v MySQL. Řádek s prefixem vyplníme podle toho, jak chceme, aby byly rozlišeny naše tabulky (například prefix „jos_“ nám zajistí, že tabulka „users“ bude v databázi nazvána „jos_users“).

V dalším kroku je nám nabízena možnost konfigurace FTP. Tu ve většině případů nepotřebujeme, zvláště pak na lokálním počítači. V následujícím a posledním kroku pak máme možnost vytvořit první uživatelský účet, který má roli administrátora. Taktéž zde zvolíme název našeho webu. Po instalaci nás program sám vybídne k odstranění instalační složky v /opt/lampp/htdocs/joomla. Toto je bezpečnostní opatření a Joomla! bez něj nebude fungovat. Pokud nelze odstranit pomocí ovládacího prvku v okně prohlížeče, musíme ji odstranit manuálně v souborovém manažeru.

4.1.2 Instalace joomla na vzdáleném počítači/hostingu

Na lokálním počítači zpravidla aplikaci vyvíjíme a testujeme. Časem chceme ale naši webovou prezentaci zveřejnit, musíme si vybrat dostatečně dobrý hosting, který bude vyhovovat svými prostředky tomu, aby na něm Joomla! běžela bezproblémově. Některé hostingy dokonce poskytují předinstalovaný redakční systém podle výběru. Pokud tak není, musíme si náš CMS nastavit a nahrát sami.

Na straně vzdáleného serveru je již vše nastaveno, jediné, co nás čeká, je překopírování souborů do souborového úložiště na straně poskytovatele hostingu, vytvoření databáze v jeho MySQL databázi a poté nainstalovat podle postupu již výše uvedeného. Soubory nahráváme přes FTP protokol. Ten nám nativně podporuje například Nautilus (souborový manažer v Ubuntu). Na straně poskytovatele získáme údaje nezbytné k přihlášení. Souborům na vzdáleném serveru zpravidla nastavujeme práva pomocí chmod 755 nebo 644. Je doporučeno zde dbát zvýšené opatrnosti při volbě hesel a uživatelských jmen, aby se systém nemohl stát obětí hackerského útoku.

4.1.3 Lokalizace Wordpress a Drupal srovnávání s Joomla

Každý CMS systém má komunitu, která pomáhá překládání do vlastního jazyka. Každý z těchto systémů má ale odlišný způsob, kterým to dělá. Například Drupal má zorganizovanou vlastní překladovou komunitu celosvětově. Jeho lokalizační balíček je rozčleněn na více podsložek rozřazených podle názvů modulů, profilů a šablon. Každá složka pak obsahuje složku translation a v ní soubor s názvem ve formátu „<nazev_souboru>.<jazykova_zkratka>.po“. Jeho obsah je formátován tak, že je určen řádek, pro který se má překládat. Určující řádek začíná zkratkou „#:“ a následuje adresa překladu v souboru. Po tomto komentářový řádek je následován „msgid“, tedy klíčovým slovem pro zprávu a k ní přiřazuje „msgstr“, tedy znakový řetězec, který obsahuje překlad. Wordpress má velmi podobný způsob řešení, ten obsahuje celkem 6 lokalizačních balíčků pro celý systém plus dva pro každé nově instalované téma(šablonu), ve dvojicích se shodným názvem, ale rozdílnou příponou. Jeden „.mo“ a druhý „po“. Balíček typu MO je takzvaný „machine object file“, který překládá PO balíčky do strojového kódu, což umožňuje daleko rychlejší fungování lokalizovaného programu za běhu.[11] Ač je to velmi

efektivní způsob lokalizace, člověk, který překládá musí rozumět tomu, jakým způsobem používat a generovat soubory, vytvářet jejich šablonu v určitém formátu a pracovat s programy k tomu určenými. Člověk mimo vývojářskou obec dle názoru autora práce nemá moc velkou šanci příjemným způsobem překládat, aniž by musel cokoliv vědět o běhu systému z pohledu programátora.

Dle názoru autora práce má Joomla poměrně jednodušší způsob překladu, protože svá data pro překlad soustředí do souborů, ke kterým je jednodušší sestrojit program, který by se soubory zacházel. Je téměř nezbytné také poukázat na fakt, že i soubory jsou jednoduššího a přehlednějšího rázu než u Wordpressu či Drupalu. Otázkou je, zda jsou konfigurační soubory s příponou „.ini“ také urychlujícím elementem, jako je tomu v případě dvou výše zmíněných systémů Wordpress a Drupal.

4.1.4 Lokalizace Joomla! pomocí lokalizačního balíčku

Systém Joomla je možno snadno překládat pomocí balíčků k tomu utvořených. Sama o sobě umí číst soubory typu ZIP, které obsahují složky rozdělené na uživatelskou a administrátorskou část. Každá složka má pak své vlastní konfiguračnísoubory a instalační soubor, který nám určuje co a kde máme v rámci instalačního balíčku uloženo. Pro překlad je nezbytná PHP funkce parse_ini_file, která konfigurační soubory s příponou .ini jednoduše rozřeže na asociativní pole a z něj pak čerpá informace o překladu na základě klíčových slov.

4.1.4.1 Soubor .ini a jeho struktura

Konfigurační soubory se skládají z hlavičky, která obsahuje informace o daném souboru a z těla. V praxi to pak může vypadat takto:

;Hlavicka – zde jsou komentáře vývojáře a tak dále.

klicove_slovo1 = „překlad1“

klicove_slovo2 = „překlad2“ ….

Hlavička je prakticky nepodstatná pro naše účely, nejdůležitější roli hrají právě klíčová slova a jejich překlady, jejich pomocí pak Joomla! zjišťuje jakým způsobem a pro který jazyk má přeložit dané klíčové slovo.

4.2 Představení aplikace JoomTee(PHP– Nette Framework)

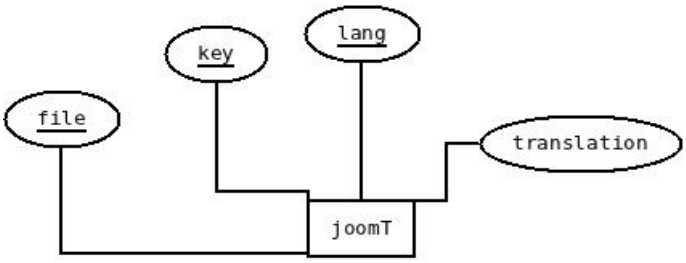
Aplikace JoomTee vzniká jako pokusná aplikace pro ulehčení budoucího překladu v rámci systému Joomla. Joomla jako taková má nyní standardizovaný (od verze 1.6 po současnost - verze 2.5.1) způsob ukládání informací o jazyku, to umožňuje vytvořit tuto aplikaci. Rozšiřuje se slovní zásoba a proto je nutné mít program, který by tuto práci usnadňoval. Samotné překládání balíčků tak říkajíc na koleni je velmi neefektivní záležitostí. Daleko efektivnější je metoda uložení dat do databáze, ze ve které by se daly případně přidávat nové překlady, nová slova by se pouze přeložila v rámci aplikace, ale autor překladu by nemusel překládat celý balíček slovo po slově. Myšlenkou této aplikace tedy je umožnit přidat nový jazyk, generovat pro něj nové balíčky a vytvářet databázi překladů, která by byla užitečná komukoli, kdo má v rámci Joomla! zájem tvořit překlady.. Aplikace bude vydána jako open source, spustitelná na lokálním počítači (Apache, MySQL, PHP).

4.3 Návrh způsobu lokalizace

Jak již bylo řečeno, systém Joomla! je nyní možno překládat pomocí externích lokalizačních balíčků, které se do ní dají snadno nahrát. Tato aplikace z toho bude vycházet a pokusí se překladatelům-vývojářům usnadnit jejich práci. Jako první věc, která přichází v úvahu (viz příloha INIprocesor) je nutnost moci nahrát a exportovat data z databáze hromadně. Získáním dat umožníme jejich využití pro překlad. Na to bylo třeba vyvinout vlastní algoritmus. Dalším krokem je tedy upravování dat v databázi, případně na základě původních nahraných moci přidávat ekvivalentní překlady tak, aby mohla aplikace generovat soubory pro každý jazyk zvlášť. Aplikace je založena na práci s nástroji z řad Open source. Mluvíme zde o skriptovacím jazyce PHP, databázi MySQL a jako testovací server byl zvolen Apache 2. Tento server je velmi rozšířen ve spojitosti právě s PHP programováním, je poměrně jednoduše nastavitelný a stabilní.

4.4 Databáze

Při řešení databáze bylo nutné brát v úvahu její rozsah, normální formy a způsoby, jakými se do ní budou moci přidávat jednoduše nové a nové jazyky, potažmo údaje o souborech, klíčových slovech v nich obsažených, jazyku a překladu. Návrh databáze probíhal ve dvou fázích. Zprvu se zdálo jako nejlepší řešení přidělit každému prvku svou vlastní tabulku. To ale nekorespondovalo s účelem jednoduššího ukládání do databáze a její přehlednosti. V rámci jednodušší operace s databází se došlo tedy k řešení ukládat prvky do jedné jediné tabulky (pracovní název „joomT“), která má 3 primární klíče a jeden atribut neklíčový. Primární klíče jsou sloupce file, key, lang. Atribut „file“ ukládá informaci o názvu souboru, „key“ poté o klíčovém slovu a „lang“ o zkratce pro jazyk. Neklíčový atribut „translation“ nám ukládá informace o uloženém překladu pro daný složený klíč. Zatímco primární klíč nemůže být nikdy stejný (stejná kombinace „file“, „key“, „lang“), atribut „translation“ nám umožňuje své opakování. U primárního klíče by bylo nežádoucí.



Obrázek 4.4.A. - Tabulka JoomT

4.5 Části aplikace JoomTee

4.5.1 IniProcesor

První částí aplikace je IniProcesor. Tato část aplikace se stará o nejdůležitější funkci celkově. Funkce, kterou zastává je export a import konfiguračních souborů. Importuje soubory tak, že je rozdělí pomocí specifické funkce v PHP a nahraje do databáze tak, aby se data roztřídila pomocí primárních klíčů. Exportní funkce pak zaručuje pro každý vybraný jazyk vlastní vygenerované soubory, které bude potřeba pro lokalizaci Joomla!.

4.5.1.1 Soubor config.php

Tento soubor nám zajišťuje přenos systémových proměnných, které potřebujeme při přihlašování do databáze. Rozliší zároveň, zda-li se nacházíme na lokálním serveru a nebo na vzdáleném veřejném serveru. Podle toho rozpozná databázi a přístupy k ní.

4.5.1.2 Soubor dbconnector.php

Soubor dbconnector.php slouží pro přímé připojení k databázi. Využívá souboru config.php k tomu, abychom do něj nemuseli ručně zapisovat konkrétní údaje pro databázi a mohli tak měnit přípojné body pouze v jednom souboru.

4.5.1.3 Soubor importData.php

Tento soubor slouží k načtení souborů „.ini“, které jsme uložili do složky „files“ , rozdělení obsažených hodnot a jejich následné uložení do databáze. K rozdělení používá funkci parse_ini_file, která dokáže hodnoty ze souboru nahrát do asociativního pole, se kterým můžeme nadále pracovat. Po nahrání do asociativního pole se zde nachází cyklus (PHP funkce „foreach“), který nám jednotlivé prvky pole uloží jako jednotné. Pomocí funkce array_push poté vytvoříme nové pole, které bude obsahovat hodnoty ukládané do databáze. To nám umožní sestavit SQL příkaz, který pomocí funkce mysql_query() uloží data do příslušných polí.

4.5.1.4 Soubor exportData.php

Tento soubor plní opačnou roli než importovací soubor, má za úkol nám podle jazyka roztřídit a vygenerovat nové konfigurační soubory, které budeme potřebovat pro lokalizaci do kteréhokoli jazyka. Nahrává data z databáze a poté je skládá. Hlavními PHP funkcemi, které jsou zde použity je fopen(),fwrite(),fclose() a následně chmod(). Chmod nám přiděluje práva pro dané soubory, je nezbytné, aby byly přístupné pro kohokoli a kdokoli s nimi mohl manipulovat, tudíž nastavíme chmod na 777. Jazyk aplikace vybíráme pomocí formuláře, do kterého zadáváme zkratku jazyka. Exportované soubory se pak uloží do složky „generatedFiles“.

4.5.1.5 Soubor index.php

Do tohoto souboru je uložen odkazový rozcestník části IniProcesor, umožní nám se dostat jak k importéru, tak k exportní aplikaci.

4.5.2 Nette Framework Formuláře

„Nette Framework je výkonný framework pro pohodlné a rychlé vytváření kvalitních a moderních webových aplikací v PHP 5. Eliminuje bezpečnostní rizika, podporuje AJAX, SEO, DRY, KISS, MVC a znovupoužitelnost kódu.“[12]

Pro tvorbu uživatelsky příjemného prostředí a efektivního upravování dat v databázi – přidávání nových a editace existujících jazyků byl vybrán Nette Framework. V současné době má na české scéně nejaktivnější komunitu programátorů a jeho funkce naprosto vyhovují záměrům pro tvorbu webové aplikace jako je JoomTee – pokusná aplikace pro překlad konfiguračních balíčků systému Joomla!. Nette Framework funguje na principu MVP(MVC) modelu a je objektově orientovaným frameworkem. MVP model se tedy skládá z Modelu, který pracuje s databází, Presenteru tvořícího ovládací prvky, zpracování a zprostředkujícího data mezi modelem a View. View je HTML šablona, kterou vidí uživatel. Nette Framework je navíc schopen existovat bez závislosti na dané aplikaci, je možné jej paralelně na do aplikací připojit aniž by člověk-uživatel, kterému se aplikaci používá (neinstaluje), cokoliv tušil nebo musel nastavovat zvlášť. To je nespornou výhodou tohoto Frameworku.

4.5.2.1 Připojení k databázi

Databáze obsahuje soubor **BaseModel.php**, ze kterého bude aplikace posléze dědit připojení k databázi. Je v něm pouze jednoduchá funkce :

```
<?php
  class BaseModel extends Nette\Object {
        protected $db;  public function __construct() {$this->db = new
Nette\Database\Connection('mysql:host=localhost;dbname=newJoomT', 'uzivatel', 'heslo');
    }
}
?>
```

Tato funkce nám definuje přípojný bod k databázi na lokálním serveru. To, že je z ní děděno, nám do budoucna zajišťuje, že nebudeme muset před každým příkazem znovu definovat přípojné body a přihlašovací údaje k databázi. To nám pomáhá nejen s úsporou kódu, ale i přidává na rychlosti aplikace.

4.5.2.2 Funkce Presenteru

Aplikace je velmi sporá, tudíž obsahuje jen jeden Presenter – HomepagePresenter, do kterého jsou zahrnuty základní funkce a takzvané „továrničky“ pro formuláře. Ty umožňují přidání nového jazyka a editaci jazyků již uložených.

Továrničky pak voláme v jednotlivých „renderovacích“ funkcích, které vytvářejí předpoklad pro vytvoření šablony. Z jednoho presenteru je možné volat více šablon, to umožňuje například urychlení stránek – přepnutím na jinou šablonu pomocí odkazu nemusíme načítat celou webovou stránku, ale například jen její část.

Formulář pro přidání jazyka umožňuje přidat celý jazyk naráz (zobrazí všechna slova a po odeslání je uloží do databáze) či překládat po jednotlivých souborech. Bude také zobrazena informace o tom, které soubory ještě přeloženy nejsou. Editační formulář nám pak umožňuje upravovat jednotlivé soubory a slova.

4.5.3 GIT

Bylo rozhodnuto umístit zdrojové kódy na GIThub.com. GIT je verzovací systém používaný při vývoji software, je zaměřen na kontrolu distribucí a jejich a správu zdrojového kódu. Git byl původně navržen a vyvinut Linusem Torvaldsem pro vývoj Linuxového jádra. V rámci systému GIThub pak můžeme sledovat historii změn v kódu, umožňuje nám stopovat celý projekt. GIT je nysní vydáván pod licencí GNU GPL verze 2.

GIThub.com je repozitář pro systém GIT, jsou na něm umístěny projekty jako je právě již zmiňovaný Nette Framework (český projekt objektově orientovaného PHP Frameworku), dfsch (vyvíjející se objektový jazyk založený na LISP), různé odnože grafického programu GIMP nebo přímo Linuxové jádro vyvíjené právě Linusem Torvaldsem.

5 Závěr

Po prozkoumání možností v rámci lokalizace „content management systému“ Joomla! (verze 1.6 a vyšší) vyvstanuly tři stěžejní problémy, které bylo nutné řešit. Prvním z nich bylo zvolení technologií pro vývoj pokusné aplikace, která bude překládat. Vzhledem k tomu, že Joomla! je psána v jazyce PHP, pro případný pozdější vývoj aplikace bylo zvoleno právě PHP. K PHP se, sice nikoli nezbytně, ale velmi často, používá serveru Apache a databázové řešení MySQL. Tato kombinace je v rámci open source systémů poměrně častá. U robustnějších a větších aplikací se pak volí spíše IIS server a MSSQL databáze a technologie ASP.NET (případně ASP.NET MVC) a jazyk C# či jiné objektově orientované nebo objektové jazyky, pro tuto aplikaci ale plně postačila open source varianta Apache, PHP, MySQL.

Druhým problémem, který bylo nutno řešit byl import dat, potažmo nalezení způsobu jak data extrahovat ze souborů původní anglické verze, aby bylo možno získat slova, která je třeba překládat. Tato slova jsou stejná pro všechny jazyky a obsahují v sobě název souboru, kterému náleží a klíčové slovo jako takové. Klíčová slova jsou v souborech se systémovou příponou „.ini“, bylo tedy nutné nalézt funkci, která by daný soubor uměla číst a pracovat s jeho obsahem. Skriptovací jazyk PHP tuto funkci poskytuje již od PHP verze 4.x a vyšší, funkce se jmenuje parse_ini_file. Daná funkce umožňuje nahrát obsah souboru do dvourozměrného asociativního pole, které nám umožní ukládat postupně data do databáze MySQL. V MySQL databázi pak bylo nutné vytvořit tabulku, která se skládala ze složeného primárního klíče a neklíčového atributu ve formě překladu. Tabulka tedy ukládala informaci o jazyku, názvu souboru a klíčovém slovu a překladu. Dvourozměrné pole se pomocí cyklu rozdělilo na jednotlivé části, které se pak do databáze uložily.

Třetím problémem, který vyvstal bylo řešení, přesného opaku od funkce parse_ini_file. Tedy nutnost extrahovat data z databáze, naformátovat a uložit do konfiguračního souboru tak, aby ho byla schopna Joomla! otevřít a přeložit.

Pro vývoj této části aplikace bylo použito čisté PHP, bez jakýchkoli nadstaveb či frameworků, ale pro druhou část bylo použito Nette Frameworku. Volba tohoto PHP Frameworku byla nasnadě právě proto, že umožňuje poměrně jednoduché a bezpečné

tvoření webových formulářů a aplikací. To, co by člověk v normálním PHP řešil například celé odpoledne, je vyřešeno již dávno předtím, než problém začal řešit, Nette Framework to obstará za něj.

Aplikace by měla být tedy schopná přeložit dané soubory, a vygenerovat nové pro nový jazyk, jejím hlavním cílem bylo zjednodušit práci při překladu a vynakládání zbytečného úsilí. Aplikace uloží daná slova a pokud v budoucnosti Joomla! vývojáři nezmění algoritmus pro překlad souborů, dá se hromadná databáze použít pro univerzální překladač tohoto systému pro práci s obsahem.

6 Seznam použitých zdrojů

6.1 Použitá literatura

1. BRAMPTON, Martin. PHP5 CMS framework development: *expert insight and practical guidance to creating an efficient, flexible and robust framework for a PHP5-based content management system*. Birmingham: Packt Pub, 2008, s. 22. ISBN 1847193579.
2. *Root.CZ*: *Open Source Software* [online]. [cit. 2012-03-27]. Dostupné z: http://www.root.cz/specialy/licence/open-source-software/
3. SMITH, Brett. *A Quick Guide to GPLv3* [online]. 2012/01/24 15:26:08 [cit. 2012-03-27]. Dostupné z: http://www.gnu.org/licenses/quick-guide-gplv3.html
4. EBENLEDER, Tomáš. Internacionalizace programů. *Linuxový seminář* [online]. [cit. 2012-03-27]. Dostupné z: http://artax.karlin.mff.cuni.cz/~ebik/nju/linuxem/i18n.html
5. *JoomlaPortal.cz* [online]. [cit. 2012-03-27]. Dostupné z: http://www.joomlaportal.cz/index.php/clanky-mainmenu-2/zaciname-s-cms-joomla/493-bart
6. *LinuxExpress*: *Redakční systém Joomla! - co je zač, pohled do historie* [online]. 4.6.2008 [cit. 2012-03-27]. Dostupné z: http://www.linuxexpres.cz/software/redakcni-system-joomla-co-je-zac-pohled-do-historie
7. RAHMEL, Dan. *Joomla!*: *podrobný průvodce tvorbou a správou webů*. Vyd. 1. Překlad Ondřej Gibl. Brno: Computer Press, 2010, 382 s. ISBN 978-80-251-2714-8 (BROž.).
8. *Drupal.org*: *History of Drupal* [online]. [cit. 2012-03-27]. Dostupné z: http://drupal.org/about/history
9. *Drupal.cz*: *O systému drupal* [online]. [cit. 2012-03-27]. Dostupné z: www.drupal.cz/o-systemu-drupal

10. *Wordpress portál - vše o redakčním systému zdarma* [online]. [cit. 2012-03-27]. Dostupné z: www.cwordpress.cz

11. *Wordpress.org*: *Codex - translating Wordpress* [online]. [cit. 2012-03-27]. Dostupné z: http://codex.wordpress.org/Translating_WordPress

12. *Nette.org*: *Hlavní přednosti | Nette Framework* [online]. [cit. 2012-03-27]. Dostupné z: nette.org/cs/hlavni-prednosti

6.2 Seznam obrázků

7 Příloha

7.1 Nejdůležitější části IniProcesoru

7.1.1 config.php

```
<?php
if ($_SERVER["SERVER_ADDR"]=="localhost"){
  define("SQL_HOST","localhost");
  define("SQL_DBNAME","newJoomT");
  define("SQL_USERNAME","root");
  define("SQL_PASSWORD","blackymo");
}
else
{
  define("SQL_HOST","localhost");
  define("SQL_DBNAME","newJoomT");
  define("SQL_USERNAME","root");
  define("SQL_PASSWORD","blackymo");
}

?>
```

7.1.2 dbconnector.php

```
<?php
include("config.php");
mysql_connect(SQL_HOST,SQL_USERNAME, SQL_PASSWORD) or die("Nelze se připojit k serveru". mysql_error()); ;
mysql_select_db(SQL_DBNAME ) or die("Nelze se připojit k serveru". mysql_error());
?>
```

7.1.3 exportData.php

```
<?php
include ("dbconnector.php");
$display = true;
if (!empty($_POST)) {
   if (strlen($_POST["lang"]) <> 5) {
        echo "Zadejte platný pětimístný formát jazykové zkratky";
   } else {
        $display = false;
        $nahraj = mysql_query("SELECT DISTINCT file FROM joomT");
echo "Generované soubory<br>";
        while ($i = mysql_fetch_array($nahraj)) :
          $path = "./generatedFiles/" . $_POST["lang"] . "." . $i["file"] . ".ini";
          $fp = fopen($path, "w");
          $v = mysql_query("select * from joomT where lang=" . '("' .
$_POST["lang"] . '")' . "AND file = " . '("' . $i["file"] . '")');
          $r = mysql_num_rows($v);
```

```
                if ($r == 0)
                    echo "Zadaný jazykový kod " . $_POST["lang"] . " neexistuje";
                else {
                    echo "Generuji v jazyce " . $_POST["lang"] . " konfigurační
soubor"."(<b>' . $i["file"] . '</b>)"." obsahující $r záznamů.<BR />";

                    while ($zaznamy = MySQL_Fetch_Array($v)):

                       $file = $zaznamy["key"] . " = " . "" . $zaznamy["translation"] . "" .
"\n";
                       fwrite($fp, $file);
                    endwhile;
                }
                fclose($fp);
                chmod($path, 0777);
            endwhile;
        }
    };
    ?>
    <? if ($display): ?>
    <form method="post" action="<? echo $_SERVER["PHP_SELF"] ?>">
      Language (vyplňte dle vzorového formátu) <input name="lang" value="en-GB">
      <input type="Submit" name="odesli">
    </form><? endif; ?>
```

7.1.4 importData.php

```
<?php
   include ("dbconnector.php");

//načte soubory ze složky ./files a rozřeže je pomocí ini parseru
   function getRecord($folder) {
         $zaznamy = array();
         $filedir = dir($folder);
         while ($soubor = $filedir->read()) {

            if ($soubor == "." || $soubor == ".." && $soubor != "*.ini")
                  continue;

            $langfilename = explode(".", $soubor);
            $lang = $langfilename[0];
            $count = count($langfilename);
            if ($count == 3) {
                  $filename = $langfilename[1];
            } else {
                  $filename = $langfilename[1] ."".$langfilename[2];
            }
            if ($filename =="ini."){
                  $filename = "ini";
```

```
            }
            $inifile = "./files/$soubor";
            $ini_array = parse_ini_file($inifile);
            foreach ($ini_array as $key => $value) {

                $zaznam = array($filename, $key, $lang, $value);

                array_push($zaznamy, $zaznam);
            }
        }
        $filedir->close();
        return $zaznamy;
    }

    $xx = getRecord("./files");
    saveRecord($xx);

    function saveRecord($zaznamy) {
        $insert1 = 'INSERT INTO joomT VALUES ';
        foreach ($zaznamy as $zaznam) {
            $insert1 .= '("' . $zaznam[0] . '","' . $zaznam[1] . '","' . $zaznam[2] . '","' .
$zaznam[3] . '"),';
        }
        $insert1 = substr($insert1, 0, -1);
```

```
$insert1 .= ";";
$insertAll = "$insert1";


echo "<br />";
mysql_query($insertAll) . mysql_affected_rows() or die("Data nemohla být v pořádku nahrána do databáze kvůli následující chybě: <br />" . mysql_error());
if (mysql_affected_rows() > 0) {
    echo "<b>Nová data úspěšně nahrána<br/><br/>(Pokud se objevil výše výpis chyb, znamená to, že byly do složky files nahrány i jiné než konfigurační soubory.<br /> Nahrání konfiguračních dat do databáze to ale neovlivnilo.)</b>";
} else {
    echo "<b>Nic nebylo změněno, data se shodují s již existujícími v databázi.</b><br/><br/>";
}
}
?>
```