ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

Bakalářská práce

Redakční a publikační systémy v prostředí WWW

Autor: Jakub Liška

Vedoucí práce: Ing. Pavel Šimek

Čestné prohlášení

Prohlašuji, že jsem tuto bakalářskou práci na téma „Redakční a publikační systémy v prostředí WWW“ vypracoval samostatně za použití uvedených zdrojů, svých znalostí a konzultací s vedoucím práce.

V Praze dne 30. 5. 2008 ..……..

Jakub Liška

Poděkování

Děkuji tímto panu Ing. Pavlu Šimkovi za trpělivost, odborné vedení a rady při zpracování bakalářské práce.

Redakční a publikační systémy v prostředí WWW

Souhrn

Obsahem práce je charakteristika redakčních systémů, koncipována se záměrem odbornějšího vysvětlení principů jejich existence způsobem, kde hloubka přínosu pro čtenáře bude záviset na jeho dosavadních vědomostech z oblasti informačních technologií. Přesto bylo vyvinuto úsilí nezanechat v práci nevysvětlené skutečnosti, které by vytvoření co nejlepší představy o redakčních systémech bránily. Literární rešerše svojí charakteristikou redakčních systémů, následovanou představením několika nejznámějších z nich, poskytne informační zázemí pro důkladný rozbor redakčního systému Habari, jeho filosofie a principů jeho funkce.

Klíčová slova

Uživatel, dokumentace, zdrojový kód, PHP, databáze, architektura aplikace, internetové aplikace, server, šablony, rozšíření, API, administrační prostředí

Editorial and publishing systems for WWW

Summary

Bachelor’s thesis focuses on characterization of publishing systems for WWW and it has been conceived with an intention of an expert explanation of principles of their existence in a way, in which the depth of experience gained by reader from this thesis will be depending on his knowledge in the sphere of information technologies. Nevertheless, significant effort has been made to leave no unexplained facts, which could prevent making the best possible conception of publishing systems. Literary research, by its characterization of publishing systems, followed by a presentation of the best known ones, provides fundamental information for profound analysis of publishing system Habari, its philosophy and functional principles.

Keywords

User, documentation, source code, PHP, database, application architecture, internet applications, server, templates, module / plugin, API, administration interface

Obsah

1 Úvod

V dnešní době komunikačních technologií už internet představuje nejen transfer dat a základní komunikaci, ale stává se zdrojem příjmů jak pro firmy, tak i pro obyčejné uživatele, kteří snadno získávají relevantní informace. Pravdou ale je, že i tam, kde lze dobře vydělat, se musí investovat. Investovat čas a peníze do tvorby webových prezentací, které nejlépe osloví a zaujmou cílovou skupinu uživatelů internetu. Zaujmout musí nejen z estetického hlediska, ale i po obsahové stránce, tedy širokým spektrem relevantních a aktuálních informací. Proces prezentace cílové skupině musí být co nejjednodušší, aby ho mohlo využívat co nejvíce lidí a výběr se nezužoval jen na ty znalé problematiky informačních technologií.

Na základě těchto skutečností se začaly vyvíjet webové aplikace, které se ztotožňují s pojmem publikační, respektive redakční systémy. V anglicky mluvících zemích nesou název „editorial and publishing systems“ neboli „content management systems“. Jejich efektivita v publikaci obsahu na internetu a také zdroj příjmů z něho plynoucí představoval v průběhu posledního desetiletí silný zdroj motivace k jejich vývoji a zdokonalování.

Ve třetím tisíciletí jsou redakční systémy základním vybavením téměř každého člověka pravidelně publikujícího na internetu. Rozšířily se na celé desítky milionů websites a tento trend je, byl, ale i bude dlouhodobě rostoucí. Důvod je prostý. S rostoucí popularitou redakčních systémů klesají nároky na technologické znalostní předpoklady k publikování na internetu, což je ve spojení s narůstající vybaveností obyvatelstva výpočetní technikou a připojením k internetu znamením o budoucí podobě a vývoji komunikace, respektive toku a zdrojů informací na internetu.

Taková kvantita informací jaká je dnes na internetu, existuje z velké části díky redakčním systémům. Bývá polemizováno, zdali to není spíše negativní efekt působící na úkor kvality, navíc informace se musí pracněji třídit a vyhledávat ty správné. V anglickém světě informací už jsou tyto otázky velmi alarmující, v českém zatím nikoliv. Přesto jsou bránou k informacím na internetu vyhledávače, které dokážou zajistit místo na výsluní těm lepším, originálnějším a užitečnějším informacím, jakési tržní prostředí, které nastoluje v té mase informací řád a spravedlnost vyzdvihující kvalitu.

Redakční systémy jsou obyčejné webové aplikace a jako takové i vznikají. Vývojem komunitami jako volně šiřitelné aplikace, tvorbou za účelem prodeje, ale i jejich vývojem na míru. Proto nejsou aplikacemi jednotné podoby, leč většinou stejné podstaty. Člověk znalý mnoha redakčních systémů při pohybu po internetu nevnímá jen obsah a vzhled website, ale i redakční systém, který mu informace prezentuje.

2 Cíl práce a metodika

Bakalářská práce poskytuje ucelený pohled na problematiku webových redakčních a publikačních systémů, ale i důkladnou prezentaci redakčního systému Habari se zaměřením na jeho funkční principy, developerskou komunitu kolem projektu a nástrojů využívaných při vývoji.

Začíná seznámením s filosofií a podstatou publikačních systémů a definováním jejich role na internetu. Neméně důležité je bezpochyby vymezení technických podmínek, které jsou vyžadovány k chodu redakčního systému. Následuje představení diferenciace redakčních systémů podle jejich specializace a uvedení několika nejvýznamnějších systémů z hlediska jejich předností a nedostatků. Cílem jejich prezentace je nalezení odpovědi na otázku, zdali mají představené redakční systémy perspektivu a budoucnost nebo pozvolna upadají v zapomnění. Nápomocné při tom bude jejich ohodnocení na základě množství funkčnosti a stanovených kritérií.

Práce se ztotožňuje jak s uživatelským pohledem na problematiku, tak i s pohledem vývojáře, především v praktické části představení projektu Habari. Pohledem vývojáře jsou myšlena hlavně fakta důležitá pro člena komunity vytvářející redakční systém Habari. Jedná se o skutečnosti, které je nutné znát pro neformální členství v komunitě, což je jednak způsob komunikace mezi členy, ale hlavně jádro celé komunity, tedy redakční systém Habari, protože důvodem její existence je vývoj tohoto systému. Jde o fázi bakalářské práce vyžadující znalost programování, která se zaměřuje na architekturu redakčního systému Habari a způsob provedení a řešení tvorby jeho podstatných částí.

Cílem je odhalení potenciálu redakčního systému Habari metodou orientovanou subjektivněji v oblasti analýzy funkcionality a spíše objektivně ve sféře použitých technologií a uplatňovaného přístupu vývoje. Mimo jiné práce směřuje k odhalení potenciálu komunity projektu Habari a nároků na její neformální členství. Projekt se nachází ve fázi dokončení jádra, což silně odlišuje postup hodnocení od systémů existujících delší dobu.

Přínosem pro čtenáře práce bez znalostí programování je především jeho seznámení s oblastí publikování na internetu prostřednictvím redakčních systémů a představení konkrétních možností. Pro čtenáře zkušenějšího v oblasti informačních technologií bude zajímavé seznámení s prostředím kolem vývoje open source[1] projektu a zasvěcení do technického provedení redakčního systému Habari a aplikovaných metodologií.

[1] *Software s volně dostupným zdrojovým kódem*

3 Charakteristiky redakčních systémů

Redakční neboli publikační systém je pojem zastupující velké množství různorodých webových aplikací. Nemusí se jednat o aplikace sloužící k publikaci článků nebo zaměřující se na redakční činnost. Obecným účelem je prezentace a správa obsahu na internetu. Mohou to být obrázky, text, videa, reklamy, animace, ale ve většině případů se jedná o publikaci článků. Proto nejsou existující redakční systémy jednotné. Kromě zaměření systému je dalším důvodem způsob vývoje, respektive volně šiřitelný redakční systém se liší od komerčního systému, obzvláště systému vyvinutému na míru například gigantickým zpravodajským společnostem.

3.1 Historie redakčních systémů

Z logického úsudku lze dojít k závěru, že prvenství ve tvorbě redakčních systémů si z hlediska data vzniku hájí mnoho tvůrců. V této oblasti nelze označit nic, respektive nikoho konkrétního, protože za prvopočátky lze považovat dobu, kdy si programátoři vytvářeli sami pro sebe webové aplikace na principu jednoduchého redakčního systému za využití CGI[2] webového serveru. Je ovšem zřejmé, že první kvalitní komerční redakční systém musel znamenat velkou revoluci, protože v dnešní době by si mnoho lidí, prezentujících jakýkoliv obsah na webu, nedovedlo tuto činnost bez něho představit.

3.1.1 Původ programovacích jazyků pro tvorbu RS

Dnes je drtivá většina redakčních systémů založena na jazyku PHP, který má svůj původ v Dánsku. Vyvinul ho Rasmus Lerdorf v průběhu let 1994-95 na základě jazyka C a Perl. V podstatě se snažil vyvinout vhodnější jazyk, který by nahradil Perl, na kterém byly postaveny jeho osobní webové stránky [1]. Proto je PHP tolik podobný jazyku Perl. Python nebo Perl byly k tvorbě RS využívány, ale tato oblast nebyla důvodem pro jejich tvorbu.

V podstatě mnoho jiných programovacích jazyků lze použít k tvorbě interaktivních webů, ale na rozdíl od jazyka PHP zde není možnost jejich snadného použití přímo v HTML[3] dokumentech. Následující rok po vzniku PHP vzešla technologie ASP[4] z dílny Microsoftu. Největší konkurent jazyka PHP v oblasti programování dynamických internetových stránek. Z toho lze usuzovat, že první redakční systémy v jazyku PHP vznikaly v druhé polovině 90. let, kdy se PHP stalo populárním. Jedním z nejznámějších RS, často označovaný jako nejstarší kvalitní open source CMS[5], je phpNuke, který vznikl roku 1998.

---

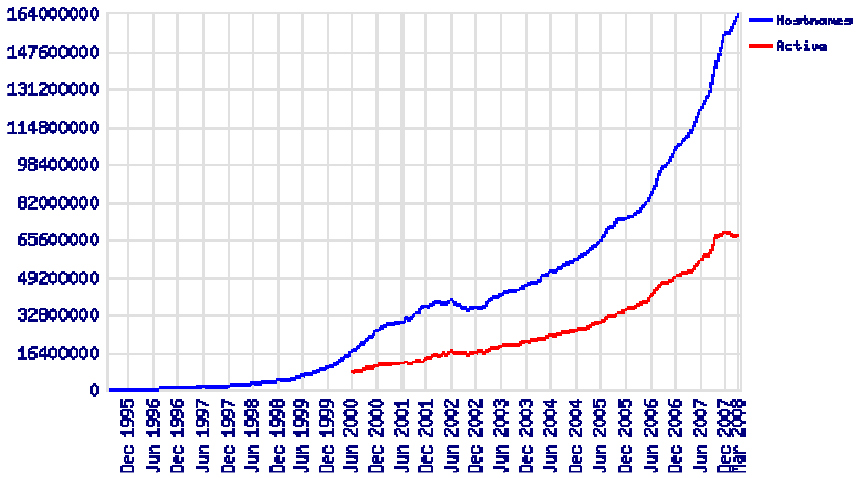
[2] *Protokol, tvořící rozhraní pro externí aplikace vůči webovému serveru*
[3] *HyperText Markup Language – značkovací jazyk, jímž se formátuje obsah html dokumentu pro přenos po internetu*
[4] *Active server pages – technologie a prostředí pro skriptování na straně serveru*
[5] *Content Management System – systém pro správu obsahu neboli redakční systém v českém překladu*

3.1.2 Důvod vzniku

Podle statistik bylo v březnu na World Wide Web téměř 70 000 000 aktivních webových sídel a kolem 164 000 000 „hostnames[6]“, čili dalších 92 000 000 potenciálních webových sídel, jejichž započtení je ovšem způsobeno skutečností, že statistiky považují subdoménu[7] jako hostname, bez ohledu na to, má-li přidělenou unikátní IP adresu nebo ne. Dále je toto číslo ovlivněno důsledkem doménových spekulací, takže lze na stránce najít např. reklamy nebo kontakt na registrátora či zájemce o doménu. V roce 2000 bylo aktivních webových stránek pouhých 14 000 000 [2].

Tento neskutečný nárůst způsobil internetový boom v kombinaci s masivním rozšířením redakčních systémů. Důkazem může být nárůst webových stránek služby Blogger[8] někdy až o milion kusů měsíčně za posledního půl roku. V České republice zaujímá jeho místo společnost Jyxo, s.r.o. se svým redakčním systémem Blog.cz. Tyto blogy se počítají do výše uvedených statistik, přestože jich pod jednou doménou existují milióny se svou vlastní subdoménou.

Z těchto poznatků nepřímo vyplývá, že redakční systémy umožní téměř komukoliv prezentovat svůj obsah na webu, což je jeden z důvodů jejich vzniku. Druhým důvodem je nepochybně skutečnost, že ušetří spoustu času lidem, kteří programovaní v PHP a tvorbu webu ovládají. Ušetří je práce s vytvářením interaktivního administračního a uživatelského rozhraní, což je nezbytnost všech websites, které neprezentují jen jeden a ten samý v čase neměnný obsah.



***Graf 1**: Vývoj počtu aktivních webových stránek a hostnames*
http://news.netcraft.com/archives/2008/03/site_count_history.gif

---

[6] *Doménová adresa, přiřazená hostitelskému PC (pojem hostnames je mnohovýznamný bez jednomyslné definice)*
[7] *Doména, která je součástí domény vyššího řádu.*
[8] *Blogger je v podstatě redakční systém společnosti Google, jehož prostřednictvím má možnost vytvořit si vlastní stránky i člověk naprosto neznalý webdesignu, který se vyhne i starostem o zajištění webhostingu a domény.*

3.2 Podstata RS

Redakční systém je v podstatě jakýmsi vzorem kvalitních, interaktivních webových stránek s obsahem odděleným od formy. Filosofii používání RS může vystihnout odpověď na otázku: Proč něco pracně vytvářet neustále znovu a znovu od základů, když už je to vytvořeno někým jiným, hodně dobře a zadarmo? Proč vytvářet interaktivní webové stránky od úplných základů, když mohu použít perfektně funkčně vybavenou webovou aplikaci pro publikaci na internetu, vytvořenou někým jiným? Ušetřený čas by mohl být využit například ke tvorbě kvalitnějšího obsahu stránek. Pro člověka, který potřebuje často publikovat nejrůznější obsah na internetu, je RS naprosto nezbytnou výbavou.

Definovat pojem redakční systém komplikuje skutečnost, že tak bývají označovány mohutné systémy jako je např. ORIS[9], ale v podstatě i website, respektive webová aplikace, vytvořená server-side[10] skriptovacím jazykem (např. PHP, ASP) s patřičnou architekturou, funkčností a důkladně odděleným obsahem od formy. Je otázkou několika hodin pro PHP programátora vytvořit jednoduchý redakční systém, ale vytvořit opravdu kvalitní systém může skupině několika programátorů zabrat půl roku.

Toto je pohled na redakční systém ze strany člověka s potřebnými znalostmi techniky tvorby webových aplikací. Tito lidé jsou si ale také vědomi, že provozovat webové stránky dlouhodobě, v řádu let, není jednoduchá záležitost. Komplikace, které se pravidelně a zákonitě objevují při používání redakčních systémů, které člověk dostatečně nezná, se velice špatně řeší. Toto je daň za možnost jejich využívání. Proto je nepsaným pravidlem, držet se jednoho nebo dvou systémů, popřípadě věnovat nějaký čas jejich studiu a všechny úpravy na jádře systému provádět s velkou rozvahou.

3.3 Požadavky na provoz redakčních systémů

I přesto, že jsou redakční systémy cestou snadné publikace na internetu i pro člověka neznalého informačních technologií, existují technické požadavky na jejich běh. Jejich zajištění sice lze delegovat na specializované osoby, ale stále jsou viditelnou překážkou ve snadné publikaci a prezentaci obsahu na internetu.

---

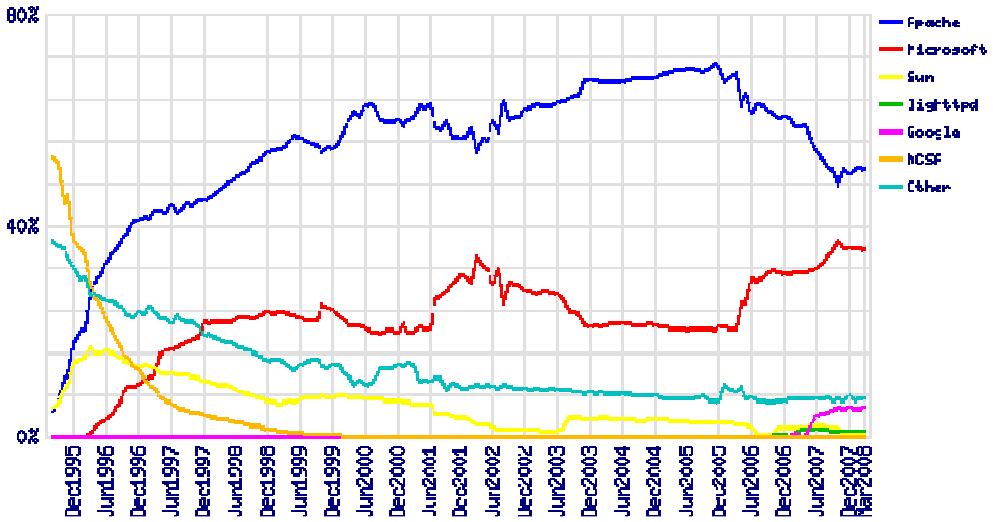
[9] *Redakční systém, na kterém jsou postaveny zprav. servery (idnes.cz, lidovky.cz atd..) mediální skupiny MAFRA*
[10] *Přívlastek, který značí, že se něco vyskytuje nebo probíhá na straně serveru*

3.3.1 Technické požadavky

Veškeré systémové požadavky na provozování redakčních systémů poskytuje většina webhostingových[11] poskytovatelů. Menší webhosting se skládá z operačního systému, serverů (http, mail, FTP, databáze)[12] a samozřejmě nesmí chybět podpora řady skriptovacích jazyků a knihoven [3]. Správce zajišťuje bezproblémový chod serverů a celého webhostingu. Je také potřeba velice rychlého a stabilního připojení na internet pro velké toky dat. Větší investice do hardwaru není na škodu, protože výpadky serveru způsobují veliké škody a rozhořčení klientů, jejichž websites nejsou na internetu zpřístupněny. Na provoz redakčního systému stačí jen webový server, databáze a interpret[13] jazyka, na kterém je redakční systém postaven.

3.3.1.1 Webový server

Redakční systémy mají prakticky stejné systémové požadavky jako většina interaktivních webových stránek. Prvotním požadavkem je webový server, který zpřístupní veškerý obsah, tzn. veškeré soubory, do World Wide Web. Webový server je v podstatě rozsáhlá aplikace, jejíž funkce je zaměřena na zpracovávání požadavků webových prohlížečů (Internet Explorer, Mozilla Firefox), ale i jiných aplikací, na konkrétní soubory. Zajistí jejich vyhledání a podle charakteru souboru následně odeslání obsahu souboru nebo výsledů interpretace skriptů, které mohou vést k mnoha dalším požadavkům. Podle protokolu http dojde k transferu dat do prohlížeče, který HTML obsah stránek interpretuje a zobrazuje uživateli [3].



***Graf 2**: Vývoj podílu HTTP serverů na World Wid Web*
http://news.netcraft.com/archives/2008/03/overallc.gif

---

[11] *Služba, nabízející zpřístupnění aplikací a webových stránek na World Wide Web.*
[12] *Hypertext Transfer Protocol (HTTP) server, File Transfer Protocol (FTP) server, mail server, database server*
[13] *Vykonavatel instrukcí programovacího jazyka*

Developer	září 07	procent	říjen 07	procent	změna %	Březen 08	procent	změna
Apache	68,228,561	50,48%	68,155,320	47,73%	-2,75	82,454,415	50.69%	-0.24
Microsoft	47,232,300	34,94%	53,017,735	37,13%	2,18	57,698,503	35.47%	-0.09
Google	6,616,713	4,90%	7,763,516	5,44%	0,54	9,012,004	5.54%	0.38
Sun	2,212,821	1,64%	2,262,019	1,58%	-0,05	1,552,650	0.95%	-0.04
Lighttpd	1,515,963	1,12%	1,541,779	1,08%	-0,04	546,581	0.34%	-0.01

***Tabulka 1**: Vývoj podílu konkrétních HTTP serverů na World Wide Web*

3.3.1.2 PHP

V dnešní době je většina stránek vytvářena pomocí jazyka PHP, díky kterému mohou být stránky nejen interaktivní (formuláře, emaily, komentáře, diskuzní fóra, statistiky), ale mohou komunikovat s jinými webovými aplikacemi a hlavně mít oddělený obsah od formy. Do HTML dokumentů se vnořují PHP skripty a soubor se pak označí koncovkou „.php[14]“, aby mohl být interpretován. Vnořené skripty mají povahu spíše vypisovací, respektive produkují výstup HTML textu pomocí funkce „echo[15]“. Za tím ale stojí desítky souborů se stovkami řádků skriptů, které musí zajistit celou logiku systému. Redakční systémy jsou prakticky celé postaveny na server-side skriptovacím jazyku, z čehož plyne další požadavek. PHP se na rozdíl od HTML dokumentů, které se interpretují v prohlížeči, interpretuje na straně serveru. To znamená, že na straně serveru musí být PHP interpret. Server pak nechá PHP soubory interpretovat a až výsledek interpretace odešle webovému prohlížeči, který interpretuje jen HTML obsah, vnořený do těchto skriptů [3].

3.3.1.3 Databáze

Třetím a posledním nepostradatelným požadavkem je databáze. Databáze zajišťuje možnost skladování dat na tabulkovém principu. Nejen díky PHP a jiným jazykům mohou být stránky interaktivní. Databáze v tom hraje svoji velice významnou roli. Data takto uložena, mimo websites, mohou být volána skripty a vkládána na libovolná místa a do libovolných proměnných podle potřeby tvůrce. Jednotlivé druhy obsahu na webových stránkách mají svoji tabulku, ve které je daný obsah uložen, a skripty vědí, kde ho najít a na co použít. Tabulky mohou být např. články, komentáře, kategorie a mnoho dalších. Nejznámější a nejpoužívanější databází je MySQL[16] od společnosti Microsoft. Hlavním důvodem je její lehká použitelnost a přítomnost snad u všech veřejných poskytovatelů hostování webů [4].

---

[14] *Koncovka souborů obsahujících mimo jiné skripty jazyka PHP, která je směrodatnou informací pro servery*
[15] *Funkce jazyka PHP, jejímž výstupem je za ní uvedený textový řetězec nebo hodnota proměnné*
[16] *Multiplatformní databázový systém, držící první místo v počtu implementací na webové servery.*

3.3.2 Uživatelské požadavky

V oblasti redakčních systémů se za uživatele obecně označují všichni lidé, kteří s ním přicházejí do kontaktu. Konkrétně se jedná o návštěvníka website, který využívá funkcí redakčního systému, registruje se, přispívá komentáři, odebírá elektronickou poštu redakčního systému apod. Uživatelem může být tvůrce obsahu, ale i správce chodu a vývoje redakčního systému. Tyto role bývají často vykonávány pouze jedním člověkem, ale u větších systémů klidně i několika desítkami. Uživatelských požadavků existuje nepochybně mnoho, ale jeden nevýlučně platí vždy a všude: jednoduchost ovládání.

3.3.2.1 Jednoduchost ovládání

Redakční systémy oslovují jak uživatele se základními znalostmi, tak i pokročilejší. Faktem je, že nelze tvořit ovládání systému podle tohoto rozdělení, protože by bylo možné rozlišovat pokročilé na ty, co programují, a začátečníky třeba na začínající uživatele internetu. První jmenovaný zřejmě bude trávit více času ve zdrojových kódech systému, začátečník se omezí na přihlášení, napsání článku a odeslání požadavku. Většina redakčních systémů do svého administračního prostředí zahrnuje co nejvíce ovládacích prvků a vsází na přehlednost, ale také na intuitivnost uživatelů začátečníků, kteří si dříve či později osvojí veškerou uživatelskou funkčnost.

3.3.3 Uživatelské role

Uživatelem je každý člověk, který vůbec přichází do kontaktu s RS. Základní rozdělení uživatelů by mohlo být na správce systému a návštěvníka webových stránek. Ale je-li systém složitější, kategorií je potřeba mnohem více a to zejména v rolích publicistů. Záleží přitom hlavně na typu webových stránek. Na některých webech lze najít takové množství znalostí, že na jejich publikaci by bylo třeba mnohem více času, než by tomu mohl tvůrce věnovat. Na vrcholu pak bývají zpravodajské servery.

3.3.3.1 Komerční důvody

Jelikož se na internetu bojuje o navštěvovanost a portály jsou financovány z reklam, nechávají majitelé webů za úplatu publikovat na svém portálu jiné autory s tím, že se jim investované peníze zhodnotí na výdělku z reklam. Takový typ portálu vyžaduje minimálně 3 uživatelské role. Role čtenáře, který má možnost okomentovat články, k čemuž je většinou vyžadována registrace. Role autora, který má uživatelská práva čtenáře, ale navíc může na portálu publikovat články. A role správce, který má práva uživatelů předchozích a navíc má práva na mazání či zamítnutí článků nebo komentářů, ale i na odebrání uživatelských práv autorů nebo čtenářů a mnoho dalších.

3.3.3.2 Organizační důvody

Redakční systémy jsou často předmětem podnikání firem. A jelikož každá firma má svoji organizační strukturu a hierarchii, uživatelské role v systému jsou naprostou nezbytností. Podobně je tomu i ve státních i jiných institucích, například na univerzitách, kde v pozici tvůrců obsahu vystupují vyučující, kteří vyžadují reakce od studentů. Tyto dvě uživatelské role vyžadují naprosto odlišná privilegia a někdy i více úrovní správců systému.

3.4 Použití redakčních systémů

Redakční systémy se dají využít mnoha způsoby. Jejich největší síla spočívá v možnosti rozšiřování website o nový obsah naprosto jednoduchým způsobem. Je-li třeba publikovat např. nový článek, stačí ho vložit v administrátorském prostředí do patřičného formuláře (redakční systém je většinou vybaven wysiwyg[17] editorem, který článek zformátuje do potřebného html výstupu), nastavit parametry jeho umístění a účelu a funkčnost RS se postará o jeho uložení do databáze a zobrazování na webových stránkách.

Manuální publikace nového obsahu na statických webových stránkách je neuvěřitelně pracná záležitost, v případě vybavenosti stránek mnohými prvky pro usnadnění orientace, kategorizaci apod., by to mohlo zabrat více času, než tvorba samotného obsahu, který má být publikován. Navíc od tvůrce je očekávána znalost syntaxe HTML a CSS[18] a mají-li být stránky kvalitní, jsou očekávány i programátorské zkušenosti. Pro používání některých redakčních systémů uživatel nepotřebuje znalosti žádné. Redakční systémy by mohly být rozděleny do několika skupin podle možnosti využití, kde je potřeba jiných druhů funkčnosti a často je obsahem jiný formát dat (videa, fotky, text, zkomprimovaná data).

3.5 Open source a komerční projekty

Mezi open source a komerčními projekty není rozdíl jen v možnosti a účelu distribuce. Liší se i samotná aplikace. V komerčních aplikacích se odráží vůle a požadavky tvůrce jednoho nebo maximálně několika. Scénář a směr vývoje open source aplikací tvoří celá komunita.

---

[17] *What You See Is What You get – metoda transformace textu i jiného obsahu dle HTML syntaxe, aby výstup na webové stránce vypadal stejně jako vstup do editoru*

[18] *Cascading Style Sheets – jazyk, vyvinutý pro snadnou změnu stylu zobrazení HTML stránek*

3.5.1 Komerční projekty

Projekty vyvíjené za komerčním účelem, které tvůrce může distribuovat za úplatu bez povinnosti poskytnutí zdrojových kódů, jsou označovány jako komerční. Mohou být ale založeny na open source řešení, vyvinutém pod licencí GNU LGPL. Uživateli přinášejí garance a řešení na míru, na druhou stranu jistou závislost na poskytovateli a finanční náročnost.

3.5.2 Open source projekty

Open source projekty jsou založeny na možnosti bezplatného používání hotového řešení včetně zdrojového kódu, který je za určitých podmínek možno i modifikovat. Zatímco obecnou funkcí licencí na software je omezení svobody ho sdílet a měnit, následující licence umožňují pravý opak.

3.5.2.1 GNU GPL (General Public License)

Pod touto licencí jsou distribuovány projekty, jejichž zdrojové kódy mohou být svobodně upravovány a používány. Další distribuce modifikovaného projektu nebo zdrojových kódů musí být opět pod touto licencí a neprobíhá-li v binární formě (s povinností autora poskytnout zdrojové kódy na požádání), musí být bezplatná [5].

3.5.2.2 GNU LGPL (Lesser General Public License)

Tato varianta umožňuje spojování a využití kódu, který není pod svobodnou licencí, např. využívání některých knihoven. LGPL stanovuje mírnější podmínky pro sestavování kódu s určitými knihovnami (např. od Free Software Foundation[19]). Takže povolení používat tyto knihovny v nesvobodných programech umožní používat svobodný software mnoha lidem. [6]

3.5.2.3 Free Software licence

Licence více omezující než výše uvedené, kde je nutno při změně zdrojových kódů umožnit další distribuci tohoto programu pod stejnou licencí (Free Software) [7].

[19] *Skupina spravující definici „Free Software“ a seznam „Free Software“ licencí*

3.6 Přehled open source CMS

Konkrétní příklady nelze všechny považovat za redakční systémy, ale spíše systémy pro správu obsahu (Content Management Systems). CMS je ve své podstatě trochu nadřazený pojem, ale v České republice se vžilo označení „redakční neboli publikační systémy“, čili překlad zužující množinu druhů pod toto označení spadající. V následujících kategoriích jsou pouze open source projekty, volně šiřitelné pod licencí GNU GPL. Až na výjimky jsou plnohodnotnými systémy pro správu obsahu. Výjimku může tvořit kategorie e-commerce a Groupware. V této oblasti mají komerční projekty velice navrch, protože jsou tyto CMS samy o sobě využívány ke komerčním účelům. Komerční tvorba CMS z této kategorie se zkrátka vyplatí. Podobně je to s kategorií „Video CMS“, kde také převažují komerční projekty. Open source systémů pro správu obsahu typu videoportálu (např. youtube.com) je velice málo a jsou nekvalitní.

CMS v daných kategoriích se svým zaměřením pouze přiklánějí k danému druhu aplikací, podle nichž jsou kategorie pojmenovány, což platí především pro Groupware, e-Commerce, e-Lerning, diskusní fóra a obrázkové galerie.

3.6.1 Portály

Portály jsou webové stránky, zaměřené na jednu nebo více tématik, publikující znalosti a informace. Jsou přizpůsobeny na reakce uživatelů buď formou komentářů publikovaných článků, diskusními fóry, emailovou komunikací přímo na stránkách, RSS[20] zdroji a mnohým dalším. Portály jsou zkrátka přizpůsobeny nejen na prezentaci obsahu, ale i na poskytování prostředí pro komunikaci uživatelů ohledně dané tématiky. Mezi nejznámější redakční systémy, které plní funkce portálů, patří: Drupal, e107, eZ publish, Joomla, PHP-Fusion, PHP-Nuke, Typo3 a desítky dalších.

3.6.2 Blogy

Weblog, v posledních letech enormní hit, jsou osobní stránky jednoho tvůrce na libovolnou tématiku s libovolným obsahem. Hlavní funkcí redakčních systémů vytvořených pro tento účel je co nejsnadnější rozšiřování obsahu webových stránek a snadná manipulace a změna designu. Autor publikuje články, názory či různá zjištění, o kterých se chce podělit. Články se obvykle řadí chronologicky. Chcete-li mít rozsáhlé stránky s kvalitním obsahem, potřebujete se soustředit a mít dostatek času jen na jeho tvorbu. Weblog je v podstatě menší portál, jen název prozrazuje, že se jedná o soukromé stránky jednoho člověka. Typickým příkaldem je Habari, Wordpress nebo redakční systém Blogger společnosti Google, či blog.cz od společnosti Jyxo.

---

[20] *Really Simple Syndication (od verze 2.0) – Technologie, založená na specifikaci XML, nabízející metodu automatického odběru informací.*

3.6.3 e-Commerce

Pod tuto skupinu se řadí CMS, které jsou přizpůsobeny pro různé formy internetové komerce. Počínaje tou nejznámější, jakou jsou e-shopy, až po stránky živící se reklamou a směrováním potenciálních zákazníků na jisté stránky, které jsou tvůrcovým partnerem a poskytují mu za tuto službu provizi. Funkce těchto CMS jsou více přizpůsobeny pro grafický obsah s textovým doprovodem. Příkladem open source CMS může být: cpCommerce, Opencart, osc2nuke, oscMax, osCommerce, phpShop nebo Zen Cart.

3.6.4 Groupware

CMS z této kategorie slouží ke kolektivní spolupráci skupiny lidí na různých projektech. Členové skupiny se často osobně neznají a jejich spolupráce spočívá na vzdálené komunikaci. Tyto CMS v sobě mohou mít vestavěné klientské aplikace, pro skupinovou textovou i hlasovou komunikaci. Například v developerské oblasti je třeba vytvářet kvalitní dokumentaci a tyto systémy poskytují každému členovi skupiny nejen možnost jejího studia, ale i její postupné tvorby bez komplikovaného přenosu dat. Součástí jsou dále např. společné kalendáře, harmonogramy, možnost sdílení dat a jiné. Takovéto CMS jsou v drtivé většině komerční a často velmi drahé, přesto velmi žádané pro jejich efektivnost ve skupinových projektech. Příkladem opensource projektů je: Achievo, Acollab, dotProjekt, eGroupWare, phpGroupWare, Streber nebo WebCollab.

3.6.5 Diskusní fóra

Diskusní fórum je nedílnou součástí internetových komunit nejrůznějšího charakteru. Aplikace tohoto typu nejsou z hlediska programování složité, důležitý je hlavně design a příjemné používání pro uživatele. Rozsáhlá diskusní fóra s desítkami, ale i stovkami tisíc uživatelů jsou také náročné na databázi. Open source CMS z této kategorie mohou být např.: bbPress, FUDforum, IceBB, Mercury, MyTopix, Phorum, phpBB, SMF, Vanilla, W-Angora, XMB.

3.6.6 e-Learning

E-learning je v posledních několika letech na výrazném vzestupu. Opět se na tom výraznou měrou podílí redakční systémy. Jeden z nejznámějších e-learningových projektů Moodle[21], je open source CMS, který používá i ČZU. Čím více se kolem takového projektu pohybuje uživatelů, tím je silnější. Například Moodle na VUT v Brně obsahuje 19223 kurzů s více než 41 300 uživateli. Moodle České zemědělské university slouží téměř 14 000 zaregistrovaným uživatelům [8].

[21] *Course Management Systém – Systém pro tvorbu výukových kurzů a systémů na internetu*

	Říjen 2007	Duben 2008
registrované stránky, využívající Moodle	35 253	41 688
kurzů	1 488 767	1 847 842
uživatelů	14 957 246	18 976 702
vyučujících	1 753 832	1 880 228
zapsaných	18 207 067	20 094 393
příspěvků ve fóru	17 039 249	22 214 919
zdrojů	10 179 253	13 384 627
kvízových otázek	12 541 319	16 711 381

***Tabulka 2**: Využití systému Moodle na celém světě*

3.6.7 Obrázkové galerie

Redakční systémy tohoto typu jsou uzpůsobeny k prezentaci obrázků. Dají se využít nejen k pohodlné tvorbě galerií obrázků, fotek apod., ale i komerčně například tak, že jejich tvůrce využívá tyto obrázky jako odkazy na partnerské stránky, kam odvádí návštěvníky, za které dostává od partnera provizi[22]. Příklady opensource RS jsou např.: Zenphoto, Plogger, Singapore, 4images, Gallery 2, PhpWebGallery a jiné.

3.6.8 Wiki

Wikipedia.org je tak významná služba, že slovo wiki začíná být s trochou nadsázky symbolem vědění. Webové stránky Wikipedia.org jsou postaveny na redakčním systému MediaWiki, který je uzpůsoben pro složitou strukturu textových dat, které wikipedia obsahuje, navzájem provázaných odkazy a dobře kategorizovaných. Jejich obsah může editovat kterýkoliv uživatel, prostý návštěvník webu. Pro psaní textu se používá wysiwyg editor, který zpracovává prostý text a definované značky. Tento redakční systém se hojně používá i v uzavřených skupinách uživatelů, většinou jako manuál, stejně jako je tomu v případě projektu Habari. Konkurenčními redakčními systémy mohou být: DokuWiki, PmWiki, QwikiWiki, UniWakka, Wikepage nebo WikkaWiki.

3.6.9 Ostatní

Tyto kategorie zdaleka nezahrnují všechny druhy redakčních systémů, ale určitě většinu druhů open source RS. Mezi open source redakčními systémy lze najít dále ještě redakční systémy pro správu dokumentů (Owl, KnowledgeTree, Flede), marketingově zaměřené CMS (SugarCRM) nebo CMS, na kterých lze postavit webové stránky, fungující jako velice schopný kalendář a organizátor (Vcalendar, WebCalendar).

[22] Affiliate marketing – Marketingová metoda založená na odměňování partnerských webů za příchod zákazníků.

4 Porovnávané redakční systémy

Vybrané redakční systémy splňují několik společných kritérií, podle kterých byly vybrány.

- Testovaná verze byla uveřejněna pod volnou licencí GPL
- Systémy jsou napsány v jazyce PHP a tudíž i server vyžaduje podporu PHP a databáze (MySQL případně dalších).
- Každý ze systémů je založen na principu modularity a architektury MVC

4.1 Joomla, Drupal, PHP-Nuke, Typo3, PHP-Fusion

Zmíněné redakční systémy jsou si velice podobné svojí univerzálností a možností zaměřit svojí specializaci určitým směrem pomocí modulů (rozšíření neboli plugin).

4.1.1 Joomla!

Název tento redakční systém přebral z významu svahilského slova „Joomla“, která znamená „všichni společně“. Vývojáři projekt plánují kompletně převést do jazyka PHP 5, namísto dosavadního PHP 4. Jomla je open source redakční systém s velkou komunitou příznivců a velkým úspěchem na poli redakčních systémů [9].

4.1.1.1 Instalace

Při prvním spuštění instalační skript automaticky zkontroluje požadavky na funkčnost a kompatibilitu prostředí. Po vyplnění potřebných informací (hostitelský web server, databázový server) se spustí konfigurační skript. Posledním krokem je manuální smazání instalačního adresáře.

4.1.1.2 Princip funkce

Tento RS je rozdělen do 3 hlavních částí. Nejdůležitější je jeho framework[23], na kterém je celý postaven, který obsahuje knihovny zvané „mambots”. Další úrovní je zastřešující třída „aplikace”, která je rodičovskou

[23] *Softwarová struktura a soubor specializovaných knihoven. Používá se jako jádro aplikace, čímž zajistí její základy, na nichž se staví konkrétní řešení.*

třídou pro mnoho ostatních, jako je mimo jiné i třetí a poslední vrstva, obsahující třídy: Components, Modules a Templates.

Inicializuje-li se soubor „index.php“, proběhne nejdříve úvodní kontrola prostředí a nastavení globálních proměnných. Následně dojde k zahrnutí šablony, kde jsou definovány bloky obsahu na stránce a jejich stylování. V bloku není žádný obsah až do doby, kdy je volána funkce „getModules(‘blok1‘)“, která zjistí, jaké moduly patří do prvního bloku a zahrne soubory, obsahující třídy pro načtení dat, jejichž instance zajistí načtení obsahu modulu s ohledem na hodnoty parametrů z URL [10].

4.1.1.3 Hotové moduly

Téměř 300 modulů je možné najít na oficiálních stránkách, všechny spadají pod licenci GNU GPL, takže jsou volně přístupné veřejnosti. Funkční škála je velice široká, pokrývající uživatelské požadavky ve všech ohledech. Zahrnuje moduly pro různé specializace webových stránek, jako jsou galerie, groupware nebo e-commerce.

4.1.1.4 Vytváření vlastních modulů

Struktura RS Joomla je poměrně složitá a hodně k tomu přispěl dlouholetý vývoj pod jménem Mambo, nicméně kvalitní dokumentace a stručné návody na vytváření modulů se na oficiálních stránkách vyskytují. Podobně jako většina vývojářů redakčních systému i Joomla postrádá základní schéma, zachycující obecný princip funkcionality, což případnou tvorbu vlastních modulů výrazně ztěžuje, protože pro vývoj těch lepších už nestačí znát jen aplikační rozhraní[24], ale důkladně chápat podstatu celého systému. Nicméně princip, na kterém Joomla funguje, je vytváření modulů velice nakloněn.

4.1.1.5 Přizpůsobení vzhledu

Na základě dodržení modularity RS a principu oddělené datové vrstvy od prezentační přišli tvůrci s možností využití šablonovacího[25] systému „patTemplate” od verze Mambo 4.5.2. Samotné šablony vzhledu se nacházejí v odděleném adresáři a mají přesně definovanou strukturu, takže při znalosti CSS se velmi lehce přizpůsobují, případně se dají vytvářet vlastní.

4.1.2 Drupal

Filosofií Drupalu je podle slov jeho tvůrce Driese Buytaerta přehlednost kódu a otevřenost zdrojového kódu. Název vzešel z Dánského slova „Druppel“, které znamená „kapat“, ve smyslu dešťových kapek. Samotný Drupal, jeho

---

[24] API (Application Programming Interface) – jedná se o rozhraní pro komunikaci dvou aplikací
[25] Web Template System – Metodologie zpracování výstupu datové a prez. logiky za vytvoření nové formy vzhledu

jádro, totiž vzešel z webové aplikace „drop.org“. Budoucí perspektiva je nepochybná, Drupal byl 22.dubna 2008 vybrán jako projekt, na jehož vývoj společnost Google věnuje 21 stipendií, ve výši $5 000 USD, v rámci projektu Google Summer of Code[26]. $105 000 USD je pro takový projekt obrovská investice a práce 21 dotovaných vývojářů se v budoucnu nepochybně projeví [11].

4.1.2.1 Instalace

Ve verzi 5 ještě instalace Drupalu vyžadovala manuální úpravu konfiguračního souboru, kde bylo třeba vyplnit údaje o databázi. Drupal 6 už je zaopatřen velice snadnou a bezproblémovou instalací. Prvně vytvořené uživatelské konto má samozřejmě automaticky veškerá administrátorská práva.

4.1.2.2 Princip funkce

Drupal organizuje veškerý obsah jako abstraktní položky. Jednotlivé položky rozdílného typu se mohou sdružovat do kategorií, rozšiřovat pomocí modulů nebo tvořit vstup pro RSS zdroj. Právě tato abstraktnost obsahu dává systému velké možnosti, základní funkce mohou být použité na různé účely [12].

4.1.2.3 Hotové moduly

Uživatelská komunita kolem systému drupal ja natolik silná, že pravděpodobnost existence žádaného modulu je a bude velmi vysoká. Jen na oficiálních stránkách lze nalézt kolem 3000 modulů, které ale nejsou všechny disponibilní kvůli různé kompatibilitě s rozličnými verzemi. Nejdůležitějším modulem je wysiwyg editor.

4.1.2.4 Vytváření vlastních modulů

Tvůrci Drupalu dbají na kvalitní dokumentaci každé verze systému. Rozsáhlou dokumentaci je možné najít přímo na oficiálních webových stránkách a po prostudování je možné vytvořit si vlastní modul, stejně tak jako pozměnit i některé součásti jádra systému.

4.1.2.5 Přizpůsobení vzhledu

Drupal, stejně jako většina redakčních systémů nabízí kompletně oddělený vzhled od funkčnosti. Programátorská jednoduchost je zabezpečena pomocí šablon „PHPTemplate”, které lze upravovat změnou CSS. Ostatní části

---

[26] GSoC - iniciativa společnosti Google, která formou grantů a dotací programátorům, zajistí stipendium pro podílení se nebo vlastním vývoji aplikací pro vybrané organizace a jejich projekty

vzhledu (např. boční lišty) je možné editovat přímo přes administrační rozhraní systému. Schválené šablony, distribuované na oficiálních stránkách, jsou na profesionální úrovni.

4.1.3 PHP-Nuke

PHP-Nuke je bezpochyby jedním z nejstarších redakčních systémů a mezi open source projekty rozhodně. Alespoň mezi těmi, co jsou používány i dnes. Proto byl za dobu jeho existence přeložen do 25 světových jazyků a domovská stránka dosahuje téměř 180 000 registrovaných uživatelů [13].

4.1.3.1 Instalace

Instalace je manuální, je třeba upravit konfigurační soubor, kde je nutné vyplnit informace o databázi. Po prvním načtení webové stránky je nutné vytvořit administrátorský účet.

4.1.3.2 Princip funkce

PHP-Nuke je typické pro dva typy přídavných modulů, které se odlišují pozicí, na které jsou zobrazené:

- Bloky – určené k zobrazení na levé a pravé straně. Poskytují funkce, které jsou přístupné na každé generované stránce (např. navigace)
- Moduly – zobrazují se v hlavní obsahové části stránky. Poskytují sadu funkcí, zaměřenou na určitou oblast využití, nezávislou na jiných modulech či blocích.

4.1.3.3 Hotové moduly

Na oficiálních stránkách PHP-Nuke je možno nalézt velké množství modulů, což je způsobeno spíše dlouhou dobou působení, než velkou oblibou v poslední době. Pomocí modulů lze systém využívat mnoha způsoby, od e-commerce nebo podnikových portálů, až po e-learningové systémy.

4.1.3.4 Vytváření vlastních modulů

PHP-Nuke je podobně jako Typo3 trochu složitější systém, co se vývoje a implementace vlastních modulů týče. Navíc jeho dokumentace nepatří k nejlepším, tudíž osvojit si principy fungování tohoto redakčního systému může být mnohem obtížnější, než se zdá.

4.1.3.5 Přizpůsobení vzhledu

PHP-Nuke nabízí přímo v základní instalaci mnoho předdefinovaných vzhledů, oddělených od funkčnosti systému a způsob, jakým se PHP-Nuke vypořádalo s otázkou grafiky, se zdá být velice jednoduchý a přehledný.

4.1.4 Typo3

O správu tohoto systému pro správu obsahu se starají dvě skupiny. Jedna zajišťuje údržbu a druhá vývoj. Z porovnávaných systémů je nejmladší a využívá ho nejmíň websites. Faktem ale je, že svojí funkčností se všem vyrovná, v případě, že je už nepředčil. Na domovské stránce je registrovaných necelých 24 000 uživatelů [14].

4.1.4.1 Instalace

Typo3 obsahuje instalační skript, který přivede uživatele ke stránce s formuláři, kde je potřeba vyplnit údaje o databázi. Přímo při instalaci se nabízí množství parametrů, které umožní přizpůsobit si systém vlastním potřebám. Po instalaci je potřeba vytvořit si administrátorský účet a nastavit určitá přístupová práva adresářům.

4.1.4.2 Princip funkce

Webové stránky, vytvořené s Typo3, mají stromovou strukturu jednotlivých sekcí, které je možné nechat zobrazovat v tomto stromě i mimo něj. Každá sekce zahrnuje stránky, jejichž obsah závisí na administrátorovi, stejně jako skutečnost, které funkce mají být pro uživatele přístupné.

4.1.4.3 Hotové moduly

Existuje množství zásuvných modulů do Typo3 s různými funkcemi. Každý modul je testovaný přímo vývojáři Typo3 a buď označený za stabilní, nebo naopak. Možné je ovšem instalovat i neotestované moduly. Typo3 obsahuje vlastního průvodce instalací modulu. Přímo na domovské website se nachází kolem tří tisíc modulů.

4.1.4.4 Vytváření vlastních modulů

Typo3 je poněkud více komplexnější RS, ale nevyžaduje hlubší znalost jeho jádra na vytvoření vlastního modulu. Na oficiálních stránkách je podrobná dokumentace k jádru stejně jako k jednotlivým už vytvořeným modulům, ale vytvořit modul se zdá být relativně jednoduché, bez nutnosti znát jádro systému. V typo3 je dokonce funkcionalita v administračním prostředí, která

vytvoří šablonu modulu a potřebné soubory podle požadavků uživatele. Dále administrační prostředí nabízí možnost hledání, stahování, zjišťování kompatibility a automatickou instalaci modulu.

4.1.4.5 Přizpůsobení vzhledu

Systém obsahuje několik zobrazovacích šablon, pomocí kterých je možná změna barev, rozměrů, typu písma a loga. Typo3 má vyvinutý vlastní šablonovací jazyk „TypoScript", který je podobný jazyku XSLT, ale poněkud vhodnější na programování v PHP a především spolupráci s Typo3. Způsob vytvoření vlastního vzhledu je v Typo3 řešen velice originálně i prakticky. Každopádně domovská website tohoto redakčního systému je velice přehledné, dobře působící a hlavně pohyb na ní je nejsvižnější ze všech srovnávaných systémů, což je dobrou vizitkou.

4.1.5 PHP-Fusion

Redakční systém, který založil a spravoval Nick Jones [15].

4.1.5.1 Instalace

Po nakopírování potřebných souborů na server je nutné přejmenovat konfigurační soubor, který je prázdný, a nastavit přístupová práva k určitým adresářům. O ostatní kroky se postará jednoduchý průvodce. Po instalaci je systém plně funkční, bez jakýchkoliv zásahů uživatele.

4.1.5.2 Princip funkce

PHP-Fusion je modulárně organizovaný a v podstatě členěný na části podle obohacení systému a pozice, na které se daná část zobrazuje (levá, pravá, obsahová). Panely jsou komponentou, která neobohacuje systém o novou funkční oblast, ale stará se o výstup některé funkce (např. aktuální počet přístupů na webovou stránku, nejnovější přidané články).

- Postranní panel – Je zobrazován na levé nebo pravé straně. Tento typ panelu se zobrazuje na každé systémem vygenerované stránce.
- Centrální panel – Je zobrazován jen v obsahové části na úvodní stránce. Dané panely jsou určené jen na zobrazování v určité oblasti webové stránky a není možné je použít na jiné pozici jako primárně dané.
- Modifikace – Jedná se o speciální část, která přímo (t.j. přepsáním souboru) modifikuje systémové jádro, a tak mění, obohacuje nebo opravuje funkčnost jádra RS.
- Infusion – Soubor funkcí, které spolu tvoří nezávislou jednotku, která rozšiřuje možnosti RS, buď pro administrační, nebo uživatelské role, případně oboje.

4.1.5.3 Hotové moduly

Oficiální stránky neposkytují dostatečnou podporu uživatelům, co se nabídky rozšíření i dokumentace týče. Zdá se být relativně obtížné zaměřit pomocí modulů website na určitou cílovou skupinu uživatelů.

4.1.5.4 Vytváření vlastních modulů

Obdobně i vývoj a implementace modulů se jeví nadmíru obtížná, díky slabé developerské dokumentaci a z toho vyplývající nesnadnosti porozumění tomuto redakčnímu systému. Oficiální dokumentace vysvětluje pomocí příkladů strukturu modulu – Infusion. Základním nástrojem na vytváření modulů a panelů je „Infusion developers kit", který obsahuje šablonu struktury panelu a Infusion, na základě které je možné vytvořit si vlastní panel či Infusion.

4.1.5.5 Přizpůsobení vzhledu

Vzhled je oddělený od funkčnosti. Existuje mnoho předdefinovaných vzhledů pro PHP-Fusion. Správné zobrazení zabezpečuje skupina tříd, deklarující povinné funkce.

4.2 Metodika hodnocení uvedených RS

Při výběru hodnotících kritérií bylo třeba postupovat s ohledem na skutečnost, že každé specifické řešení vyžaduje vlastní subjektivní zhodnocení se zřetelem na cílovou skupinu uživatelů. Hodnocení výše uvedených redakčních systémů bylo znesnadněno absencí grafických schémat a obecných principů základní funkčnosti na oficiálních stránkách tvůrců.

4.2.1 Základní vybavení

Kolik funkcí a vybavení obsahuje hodnocený RS ve své základní instalaci. Jak kvalitně je přizpůsobitelný různým specifickým scénářům, respektive typům webových stránek.

4.2.2 Přídavné moduly

Kolik modulů, resp. přídavných funkcí je dostupných, v jaké kvalitě a jak složitá je jejich instalace. Kritérium dostupnosti nejčastěji používaných modulů pro běžné webové stránky je zohledněno také.

4.2.3 Flexibilita vzhledu

Zde je hodnocena složitost modifikace vzhledu systému a jeho přípustná možná změna. Taktéž je brán ohled na stupeň náročnosti vytvoření nového vzhledu. Důraz je kladen na originálnost řešení, kterého bylo využito na vzhledovou část systému.

4.2.4 Preciznost zdrojového kódu

Tento bod zohledňuje přehlednost zdrojového kódu, například pomocí návrhových vzorů[27], buď vlastních, nebo určených standardem. Zdali jsou řádně a dostatečně okomentované jednotlivé úseky kódu a zdali je systém plně modulární, je stejně důležité jako precizní oddělení jednotlivých vrstev. Podstatnou roli hraje také kvalita dokumentace jak uživatelské, tak developerské.

4.2.5 Jednoduchost ovládání

Toto kritérium hodnotí, jak je systém řešený v souvislosti s komunikačním prostředím s uživatelem. Důraz je kladen na nenáročnost i pro méně zkušené uživatele. Všeobecně platí, že čím je systém větší, tím je toto kritérium důležitější. Ovšem snadnost ovládání by neměla být na úkor omezené funkčnosti.

4.2.6 Stabilita a vývoj

Open source projekty jsou založené na dobrovolné účasti programátorů, jejichž zručnost a zkušenost se často velmi odlišuje. To může způsobit, že se v systému vyskytují chyby. Proto je důležitá neustálá aktivita vývojářů na projektu, aby byla zajištěna stabilita a oprava chyb.

4.2.7 Dodržování standardů

Je nutné, aby RS splňoval standardy W3C[28] a taktéž obsahoval aktuální syntaxi programovacího jazyka. Nedodržení tohoto kritéria by mohlo vést k omezením kompatibility v budoucnosti.

---

[27] *Design patterns – Představují obecné řešení problémů, které se opakovaně objevují při návrhu softwaru.*
[28] *World Wide Web Consortium – mezinárodní konsorcium pro vývoj www standardů*

4.3 Zhodnocení redakčních systémů

Uvedená kritéria jsou u každého RS hodnocené body od 0 (nesplnění kritéria) do 10 (splnění kritéria v plném rozsahu). Získané body jsou uvedené v následující tabulce.

	Joomla!	Drupal	PHP-Nuke	Typo3	PHP-Fusion
Testovaná verze	1.5RC3 Takriban	4.7.8	4.8.1	4.1.6	6.01.13
Licence	GPL	GPL	GPL	GPL	GPL
Web	joomla.com	drupal.org	phpnuke.org	typo3.com	php-fusion.co.uk
Web s moduly	joomla.com	drupal.org	karakas-online.de	typo3.org	Phpfusion-mods.com
OS	libovolný	Windows/Linux	Linux/Windows	libovolný	libovolný
Databáze	MySQL 3.23.x(5)	MySQL 4.1(5) / PostgreSQL 7.4	MySQL 4.1 / PostgreSQL	MySQL,MSsql, Oracle,PostgreSQL	MySQL,MSsql, Oracle, PostgreSQL
Interpret	PHP 4.3.x(4.4.7)	PHP 4.3.5(5.2)	PHP 4.1	PHP 4.x(5.x)	PHP 4.x(5.x)
HTTP server	Apache 1.3(2.2)	Apache 1.3(2.2) / IIS5(6)	Apache 1.3 / IIS5	Apache 1.3 / IIS5	Apache 1.3 / IIS5
„Powered by …“ on Google[29]	1 270 000	3 670 000	8 630 000	50 200[30]	757 000
	Přítomnost funkčnosti v RS: A – ano, v základní instalaci, N – ne, M – jako modul				
Lokalizace rozhraní	**A**	**A**	**A**	**A**	**A**
Multi-jazykovost	**M**	**A**	N	**A**	**A**
Podpora CGI skriptů	N	**A**	**A**	N	N
RSS	**A**	**A**	A	**A**	**A**
Profilování uživatele	**A**	**A**	N	**A**	N
Správa reklamy	**A**	**M**	**A**	**M**	N
Správa zisku	**A**	**A**	N	**M**	N
Správa šablon	**A**	**A**	**A**	**A**	**A**
Úprava obrázků	**A**	**M**	N	**A**	N
Intuitive URL	**A**	**A**	N	**A**	N
Macro jazyk	**A**	**M**	N	**A**	N
Uživatelské skupiny	N	N	N	**A**	**A**
Wysiwyg editor	**A**	**M**	**A**	**A**	**A**
Verzování	**A**	**A**	N	**A**	N
SSL[31] kompatibilní	N	**A**	N	**A**	**A**
SSL přihlášení	N	N	N	**A**	N
LDAP autorizace	**M**	**M**	N	**M**	N
NTLM autorizace	N	**M**	N	N	N
Sandbox	N	N	N	**A**	N
Log událostí	**A**	**A**	N	**A**	**A**
	Bodové ohodnocení vycházející z testování těchto redakčních systémů				
Základní vybavení	8	8	3	10	5
Přídavné moduly	7	10	7	7	4
Flexibilita vzhledu	8	8	8	8	7
Preciznost zdr. kódu	7	8	6	6	5
Jednoduchost ovl.	6	6	7	6	7
Stabilita a vývoj	10	10	6	7	6
Dodržování standardů	7	7	6	8	7
	53	**57**	**43**	**52**	**41**

***Tabulka 3**: Vyhodnocení porovnávaných redakčních systémů*

[29] *Počet výsledků vyhledávání fráze “Powered by …“ vyhledávačem Google, což má velkou vypovídací hodnotu o počtu webů na internetu, používajících tento RS*

[30] *Číslo je zkresleno skutečností, že v mnohých websites postavených na Typo3 není tato fráze vyznačena formou textu, ale obrázku*

[31] *Secure Socket Layer – protokol, zajišťující zabezpečenou komunikaci a přenos dat na internetu*

4.4 Shrnutí a zhodnocení prezentovaných redakčních systémů

Každý z těchto redakčních systémů je výsledkem odhodlání tisíců lidí tvořících komunitu kolem daného projektu. Jestli splňuje požadavky dnešního uživatele a jestli má budoucnost, závisí na velikosti tohoto odhodlání, na potenciálu dané komunity kolem projektu a na jeho schopnosti oslovit co nejvíce lidí. Tyto skutečnosti se odrážejí i ve výsledcích hodnocení systémů. Drupal a Joomla jsou v současnosti středem pozornosti, vyhrávají ocenění a soutěže a na rozdíl od PHP-Nuke a PHP-Fusion nepůsobí vyprahle, ale velice perspektivně. Jejich komunita je den za dnem silnější, na domovské stránky je odkazováno čím dál více a počet stažených instalačních balíčků exponenciálně roste. PHP-Nuke ani PHP-Fusion nejsou špatnými systémy, ale pod tíhou konkurence nemají takovou šanci získat uživatele, který většinou vybírá ze dvou nebo tří nejlepších. Typo3 je výjimečný redakční systém hlavně kvůli funkčnostem v oblasti autorizace, zabezpečení i jiným, které se v ostatních redakčních systémech nevyskytují. Navíc jeho framework je používán i pro intranetová řešení, což ho staví do perspektivní pozice. Trochu vyčnívá mezi konkurencí a působí originálním dojmem.

Redakční systém Habari, představený v následující části práce, nemohl být do porovnání zařazen, protože se se svou krátkou dobou vývoje nemůže ostatním redakčním systémům rovnat, hlavně po stránce počtu modulů a vzhledů. Nachází se na počátku vývojové éry, která teprve rozhodne o jeho budoucnosti.

5 Redakční systém Habari

Redakční systém Habari je open source projektem, jehož filosofií je budoucí jednoduchá, pohodlná a efektivní publikace a správa obsahu na internetu. Tvrdá, neuspěchaná a promyšlená práce stojící za projektem, činní tento systém dostatečně věrohodným. Otázkou zůstává, jestli v sobě skrývá takový potenciál, který by z něho udělal soupeře obrovské konkurenci dnešní doby. Zdali jsou použitá řešení, s důrazem na logiku a budoucí racionálnost, opravdu výjimečná, nebo úsilí a času nebylo vynaloženo dostatek.

Redakční systém Habari se nachází ve fázi dokončení jádra, a proto nelze posuzovat jeho kvality a perspektivu jako u systémů existujících celá léta. Lze vycházet z charakteru komunity a jejích technologií používaných při vývoji aplikace. Hlavním determinantem je ovšem kvalita a originalita využitých řešení, jež by neměla být náchylná na plynutí času, technologických nároků a požadavků uživatele.

5.1 Historie a důvod vzniku

Habari je open source redakční systém, založený koncem roku 2006. Název má svůj původ ve svahilském slově „Habari“, které znamená: „co je nového.“Vznikl z iniciativy skupinky několika hlavních vývojářů redakčního systému Wordpress[32]. Spoluzakladatel publikačního systému Wordpress se začal příklánět a tíhnout ke komerčním úmyslům s projektem. To je samozřejmě zradou vůči komunitě, která se na open source projektu celé roky podílela. Není to ovšem hlavní důvod, proč výše zmínění zakladatelé projektu Habari začali od úplných základů s novým vývojem. Byli také motivováni snahou vytvořit kolem redakčního systému Habari novou, lepší komunitu vývojářů, kontributorů a uživatelů. Přijímat nové nápady a vytvářet redakční systém originální a efektivní cestou [16].

5.2 Použité technologie

Dalším, silnějším důvodem vzniku projektu Habari je skutečnost, že podobně jako mnoho jiných redakčních systémů i Wordpress přestává být atraktivním, protože je postaven na zastaralé verzi PHP4, přičemž tato verze ještě nebyla plně objektově orientovaným jazykem. Redakční systém Habari je napsaný čistě jazykem PHP5, což má pozitivní vliv na jeho architekturu. Kromě

[32] *Nejpoužívanější redakční systém na světě*

programovacího jazyka redakční systém využívá ještě technologie Javascript a Ajax.

Javascript je skriptovací jazyk, který je uzpůsobený pro poskytnutí interaktivity webových stránek, co se grafického uživatelského rozhraní týče. Ajax[33] využívá technologie Javascript mimo jiné i k vytváření požadavků na server, aby mohl objekt požadavku, přijmutý pomocí značkovacího jazyka XML nebo protokolu http, zobrazit bez potřeby znovu-načtení webové stránky. Obě technologie se v redakčním systému Habari využívají především v administračním prostředí [17].

5.3 Uživatelský pohled na redakční systém Habari

Uživateli redakčního systému Habari jsou dvě skupiny, představující dva protipóly publikujících a čtenářů. Publikující využívají jeho platformu pro publikaci obsahu na internetu a čtenáři jeho funkčnosti pro získání informací, které se snaží publikující předat. Na základě této skutečnosti je uživatelské rozhraní rozděleno na část publikační a administrační.

5.3.1 Administrační rozhraní

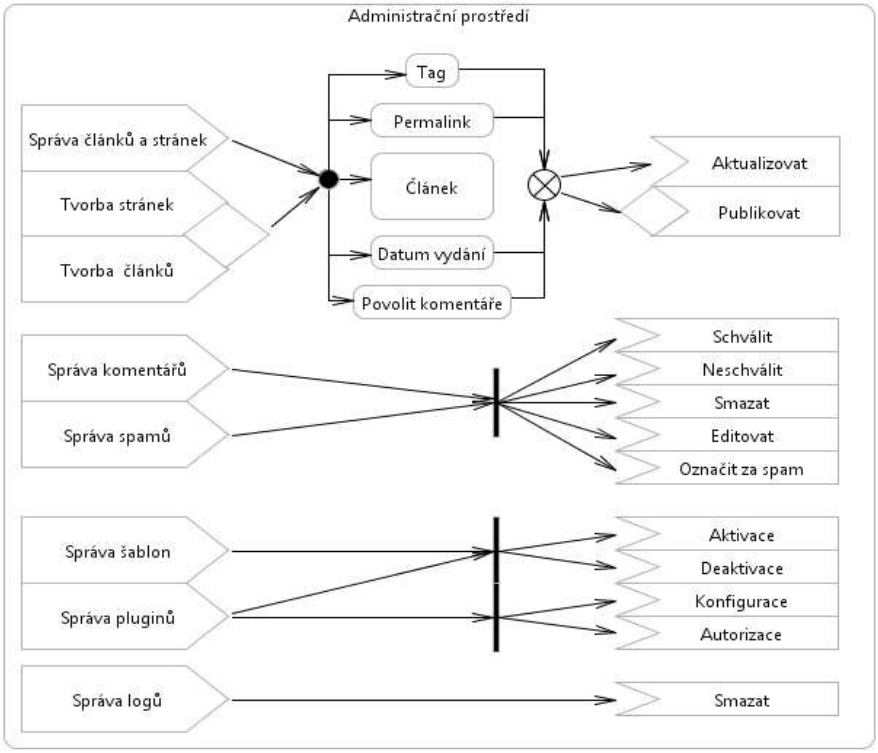
Habari je publikační systém pořád spíše ve fázi vývoje, takže z hlediska privilegií nabízí pozici správců a publikujících pouze na stejné úrovni. Všichni správci jsou si tedy rovni a mají k dispozici stejné funkce.

Administrační rozhraní poskytuje uživateli prostředí k publikaci informací a obsahu na website a zároveň k její správě a zajištění jejího bezproblémového chodu. K zajištění těchto potřeb obsahuje administrační prostředí, kromě možnosti nastavení základních parametrů systému, správu vzhledu, rozšíření, uživatelů či záznamu událostí. Jelikož redakční systém nabízí i možnost komunikace návštěvníků website s ostatními i s publikujícím, nechybí v prostředí administrace ani správa uživatelských komentářů nebo možnost likvidace spamů, čili pokusů využít této příležitosti k šíření nevyžádaného obsahu.

Klíčovým prvkem je wysiwyg editor, který umožňuje uživateli pohodlnou a rychlou publikaci obsahu na website bez potřeby znalosti HTML kódu. Editor transformuje text i jiný obsah dle HTML syntaxe, takže výstup na webové stránce by měl být identický jako vstup přijmutý editorem. Daní za používání tohoto způsobu editace je nadbytečná produkce opakujících se znaků a tagů[34], takže HTML kód je na výstupu velice objemný a syntakticky nesprávný.

---

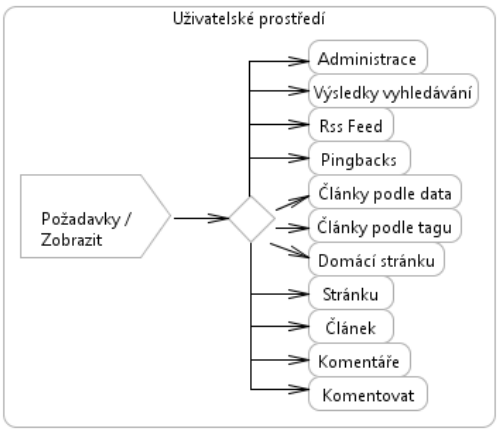
[33] *Asynchronous JavaScript and XML*
[34] *HTML tag – znak, značící začátek a konec elementu v HTML syntaxi*



Obrázek 1: *Znázornění uživatelských možností v administračním prostředí, vyvolávajících události (obrázek vytvořen v Eclipse)*

5.3.2 Publikační rozhraní

Publikačním rozhraním rozumíme prezentaci webových stránek návštěvníkům website. Jeho smyslem je podat návštěvníkům informace a veškerý obsah přehledným, srozumitelným i jiným způsobem podle libosti publikujícího nebo správce redakčního systému. Jinak řečeno tato část má za úkol přehledně zobrazovat data, která byla do systému zadána v administrační části.



Obrázek 2*: Znázornění uživatelských požadavků v publikačním rozhraní (obrázek vytvořen v Eclipse)*

5.4 Project Management

Na vývoji rozsáhlých aplikací se podílí různou měrou mnoho programátorů. Není podstatné, jedná-li se o open-source projekty, na kterých spolupracují lidé po celém světě nebo vývoj v rámci intranetu developerské společnosti. Obojí vyžaduje vysokou úroveň koordinace, kvalitní komunikační prostředí a datovou správu projektu. Na vývoji redakčního systému Habari se podílí zhruba 15 vývojářů, kteří se kromě několika zakladatelů, žijících na jihovýchodě USA, většinou osobně ani neznají, protože žijí po celé Evropě, Austrálii, či ve zbytku světa.

5.4.1 Internet Relay Chat

Pilířem komunikačního prostředí skupiny vývojářů projektu Habari je nepochybně Internet Relay Chat (IRC), který poskytuje ideální možnost debaty v reálném čase [18]. Debaty, ve které jsou vítáni jak uživatelé redakčního systému, tak vývojáři se svými návrhy na zlepšení. Na IRC kanálu #habari serveru „Freenode“ lze nalézt podporu 24 hodin denně, díky rozptýlení vývojářů po celém světě. Tvůrci rádi reagují na prosby o radu či návrhy na vylepšení, protože redakční systém Habari se nachází ve fázi dokončení jádra a intenzivního vývoje nadstavby, vyžadující inspiraci uživatelem.

Skupinová diskuse a debata o dané problematice, kterou IRC nabízí, vyniká svou jedinečností konfrontace mnoha názorů a pohledů na věc, jenž vede k využití toho nejlepšího možného řešení a zabránění implementace chybných rozhodnutí z důvodu absence znalostí, jimiž disponuje jiný vývojář. Jelikož předmětem debaty je problematika, ve většině případů vyžadující podloženost zdrojovým kódem, využívají diskutující služby, rozšiřující možnosti Internet Relay Chatu. Jedná se především o specializované webové aplikace, nabízející úložiště pro textová data, jejich zformátování do vhodného tvaru a zpřístupnění na internet, velice flexibilním způsobem[35]. Člen diskuse má příležitost takto zpřístupnit problematický zdrojový kód ostatním během několika vteřin.

5.4.2 Version Control System

Při zrodu projektu Habari se tvůrci rozhodli využít pro jeho správu společnosti Google. Jedna z jeho služeb GoogleCode[36] nabízí relativně kvalitní hosting pro open source projekty. Důležitou součástí tohoto hostingu je správa projektové dokumentace, diskusní fórum, Issue Tracker[37] a mimo dalších především správu verzovacím systémem Subversion. Spojení těchto služeb představuje pro správu projektu, jakým je Habari, dostačující prostředí. Version

[35] *http://pastebin.ca, http://pastebin.org http://pastoid.com http://pastie.textmate.org*
[36] *Webové služba společnosti Google, hostující open source projekty a jejich zdrojové kódy a seznam aplikací Google se zveřejněným aplikačním rozhraním.*
[37] *Aplikace, poskytující časový sled záznamů změn, provedených ve zdrojovém kódu*

control system (VCS), neboli Source Control Management (SCM), je naprosto nepostradatelnou součástí při vývoji softwaru.

Jedná se o způsob zaznamenávání změn digitálních informací, který vzešel v 60. letech a způsobil velkolepý převrat ve vývoji aplikací, což platí dvojnásob pro velké open-source projekty, jejichž hojnost v 90. letech byla velice pozitivně ovlivněna nejen internetovým rozmachem, ale také díky průkopnickému „versioning“ systému Revision Control System, ze kterého vzešel asi nejznámější Concurrent Version System. Ten se v posledních letech dostal do stínu novějšího verzovacího systému Subversion, jenž je využíván projekty typu: Apache, Python, Ruby, ale i společností Google pro open-source projekty, jimž poskytuje hosting.

5.4.2.1 Subversion

Subversion je založeno na bázi client/server architektury. Důležitou roli hraje centrální sklad dat na straně serveru (repository), na který mají přístup klienti, kteří zapisují do souborů, mají-li na to uživatelská práva, nebo soubory mohou jen číst. V podstatě se jedná o obecný princip „file serveru“[38] s tím, že je zaznamenána každá provedená změna v celém souborovém systému. Klient má zkrátka možnost vidět nejen poslední stav, ale i všechny předchozí, spolu s doplňujícími informacemi modifikace (kdo, kdy, proč).

Z kapacitních důvodů a z hlediska efektivity data nejsou duplikována. Tato skutečnost je zajištěna tzv. diff[39] programy. Stačí zaznamenávat pouze diff výstupy (patch[40]), kterými je původní nebo naopak nejnovější soubor modifikován do stavu (v časovém kontinuu vývoje), jaký vyžaduje klient.

V podstatě je na tomto principu založen obyčejný upgrade programů (nekompilovaných) nebo rozšíření o novou funkčnost. Soubor, kterým se upgrade provádí, není nic jiného než kompilace souboru výstupu diff utility a zdrojového kódu programu, který zajišťuje modifikaci objektu, respektive zanesení změn (výstupu diff utility) do programu.

Výhodou Subversion je skutečnost, že vzdálený přístup do repository je klietovi zprostředkován protokolem HTTP. Vývojářům projektu Habari stačí mít nainstalovanou klientskou aplikaci Subversion a své autorizační údaje k tomu, aby mohli provádět správu souborů. Na začátku roku 2007 ale nebyly hostingové služby GoogleCode pro open source projekty na takové úrovni jako dnes a vývojáři projektu Habari se rozhodli přemístit repository pod SVN na svém vlastním oficiálním serveru[41] [19].

---

[38] Server spravující uskladněná data a poskytující jejich zpřístupnění ostatním stanicím pomocí protokolů přes síť
[39] Původně unixová utilita pro porovnávání dat, jejímž výstupem je pouze rozdíl mezi dvěma soubory a právě tento rozdíl je hlavní element verzovacích systémů.
[40] Program, který podle instrukcí mění textové soubory
[41] http://svn.habariproject.org/habari/

5.4.3 Bug Tracker

Třetí a možná nejdůležitější součástí, kterou komunita kolem projektu Habari od nového roku využívá, je Trac Project[42], vyvinutý společností Edgewall software. Ta se nechala inspirovat programem CVSTrac, který reimplementovala v jazyce Python. Trac v podstatě prohlížečem využívá Subversion repository a svým elegantním grafickým prostředím poskytuje uživatelům a vývojářům Habari jednak příjemné ovládání a přístup k souborům, ale i velice přehledný pohled na všechny okomentované změny, provedené v minulosti, zdrojové kódy, poskytne správu dokumentace a grafické, statistické i jiné funkce, jaké by měl dobrý „Project management“ software nabízet[43] [20].

5.4.4 Správa dokumentace zdrojových kódů

Pro správu dokumentace zdrojových kódů se vývojáři rozhodli využívat open-source projektu PhpDocumentor, který vyvinul Joshua Eichorn a široká skupina kolem něho. Tento nástroj zajišťuje autodokumentaci[44]. Stačí jen tvořit při vývoji komentáře podle jisté syntaxe a PhpDocumentor je dokáže vyextrahovat spolu s informacemi o souborech, třídách, funkcích i jednotlivých proměnných do námi zvoleného formátu (.chm, .pdf, .html, .xml).

Psát obsah komentářů podle standardů PhpDoc má i další výhody v určitých vývojových prostředích jako je např. Eclipse[45]. Týkají se interaktivních informací, které nám jsou k dispozici v celém projektu o funkcích, třídách apod., nadefinovaných v komentáři nad danou funkcí nebo třídou, při jejich deklaraci. Mnohem důležitější je ale skutečnost, že takto provedená dokumentace je na rozdíl od samotného zdrojového kódu schopna poskytnout určitým aplikacím vypovídací schopnost o vnitřní logice a lze ji tedy využít jako vstup do těchto aplikací, jež logiku modelují do grafické podoby (grafy, UML, apod.). [21]

5.4.5 Uživatelská dokumentace

Další neodmyslitelnou součástí projekt managementu je uživatelská dokumentace, případně informace pro potencionální kontributory, kteří by se chtěli podílet na vývoji redakčního systému Habari. Využít pro to redakční systém MediaWiki, na kterém je postavena Wikipedia, se zdá být nejrozumnějším řešením, hlavně co se týče rozšiřování dokumentace do budoucna.

---

[42] *Bug Tracker / Issue Tracker - http://trac.edgewall.org/*
[43] *http://trac.habariproject.org/habari*
[44] *Metoda, díky které po tvorbě zdrojových kódů a jejich komentářů, odpadá tvůrcům povinnost pracně tvořit developerskou dokumentaci.*
[45] *Nejrozšířenější open source developerská platforma (vývojové prostředí), napsaná v jazyce Java*

5.5 Work Flow

Work flow znamená v případě webové aplikace souhru základních procesů, které se inicializují při uživatelském požadavku na systém, soubor výkonů, které se musí zařídit. Na principu laviny, kdy uživatelský požadavek vyvolá jeden proces, který spustí další a další, se vykonají všechny náležitosti požadavku.

5.5.1 Inicializace uživatelem

Po odeslání požadavku uživatelem, respektive jeho klientskou aplikací (browser) na server, obsluhující redakční systém Habari, mohou nastat dvě situace. Jde-li o požadavek na konkrétní soubor (audio, obrázky, CSS apd.), server jej vyhledá a doručí bez zapojení redakčního systému do procesu. V opačném případě je požadavek serverem přesměrován na soubor index.php a dojde tak k inicializaci celého systému. Tenhle způsob inicializace systému lze nalézt v naprosté většině redakčních systémů.

Úkolem souboru index.php je tedy připravení systému na chod a zajištění zpracování a vykonání požadavku. Soubor index.php je navržen tak, aby veškeré operace a procesy měly co nejnižší odezvu a systém byl flexibilní a svižný. V průběhu vývoje je jedním z nejvíce se měnících souborů v celém projektu, protože je kladen velký důraz na originalitu a transparentnost řešení.

V prvním kroku interpretace souboru dojde k ověření, zdali je interpret jazyka PHP vyšší nebo roven aktuální požadované verzi (PHP 5.2.x). Dále je v souboru zajištěno, aby patřičné proměnné nabyly konstantní hodnoty (cesty k souboru index.php) podle umístění RS v souborovém systému serveru.

5.5.2 Output Buffering

Poté následuje aktivace output bufferu[46], jež uchová veškerý výstup skriptu v paměti serveru. Primárním účelem je zamezení průběžného odesílání výstupu klientovi, což není vhodné v případě vzniklých komplikací. Výstup bude odeslán až na konci interpretace v případě, že všechno proběhne v pořádku.

5.5.3 Významná role funkce __autoload()

Komplikace v rychle se vyvíjející aplikaci, jakou Habari je, často vznikají z důvodu nedostatečného definování tříd a objektů a jejich souborů, měnících své umístění. Vyhnutí se této skutečnosti je v projektu zajištěno funkcí __autoload, která automaticky vyhledá a načte do proměnné soubory, obsahující jednotlivé třídy. Pomocí této funkce je v projektu vyřešena problematika

[46] *Oblast paměti, využívána k dočasnému uložení dat, předtím než budou dále zpracovávána.*

rozptýlení tříd (souborů, stojících za jednou třídou) do více adresářů v průběhu používání systému uživatelem.

Při úpravách systému je totiž praktičtější nezasahovat do původních souborů, ale upravovat kopie s následným umístěním do adresářů, které jsou pak při procesu výběru zdrojového adresáře načítání tříd preferovány před originálními. Stejně tak se do nich budou ukládat nově přibývající třídy pluginů, schémat apod. Vezmeme-li v potaz, že redakční systém bude spravovat desítky websites najednou, originální řešení, což je jedno z hlavních kréd projektu Habari, jsou velice vítána.

5.5.4 Orientace v souborovém systému

Orientace v široké adresní struktuře redakčního systému i celého serveru je řešena v jedné z hlavních tříd „Site“, kde jsou deklarovány proměnné, nabývající hodnot adres, respektive cest ke specifickým souborům, přičemž rozhodující je, spravuje-li systém jednu nebo několik websites.

5.5.5 Report chyb

V souboru index.php nechybí ani funkce reportu chyb „error_reporting()“, za využívání třídy „error“. Funkce určuje, které chyby budou nahlášeny a dostupny správci serveru v záznamech („logs“). Bez této funkce je totiž velmi obtížné zjistit, kde a proč nastal problém. Funkce spočívá ve vylepšení záznamů serveru o nastalých komplikacích.

5.5.6 Spojení s databází

Dále následuje ověření, zdali je systém nainstalovaný nebo není. Instalací se vytvoří soubor „config.php“, ve kterém je definováno DSN[47]. Spravuje-li redakční systém více websites, vytvoří se instalací soubor „config.php“ ke každé z nich. Existuje-li soubor „config.php“, znamená to, že instalace proběhla a systém se pokusí spojit s databází. V případě, že spojení selže, proběhne instalace znovu. V každém případě dojde k odeslání hlavičky pomocí funkce „header()“ a nastavení jazyka, používaného systémem na zobrazování generovaných zpráv.

5.5.7 Načtení pluginů

V další fázi přichází na řadu načtení a inicializaci pluginů[48]. V případě projektu Habari se dělí tato rozšíření na systémová, která jsou vyvinuta a ověřena hlavními vývojáři a zastávají důležitou funkci. Spadají pod správu

[47] *Database Source Name - definování údajů, nutných pro spojení s databází*
[48] *Rozšíření funkcionality systému*

vývojářů a podléhají úpravám a inovacím s vývojem nových verzí projektu Habari. Podobně jako u konkurenčních redakčních systémů je možnost jejich aktivace a deaktivace velice jednoduchá a náleží administračnímu prostředí.

5.5.8 Zpracování uživatelského požadavku

Až po uskutečnění všech předchozích operací přichází řada na zpracování uživatelova konkrétního požadavku. Publikační systém Habari je navržen podle vzoru MVC (Model-View-Controller), kde Model reprezentuje data a logiku, která z nich dělá informace. View zařizuje zprostředkování dat do formy vhodné pro prezentaci a interakci s uživatelem. Controller zpracovává a reaguje na uživatelské i jiné události a vyvolává změny v modelu a prezentaci.

5.5.9 Automatické, časově řízené funkce

V neposlední řadě existuje pro zkušenější uživatele možnost nastavení automatických funkcí, jako je např. pravidelné obnovování obsahu, odstraňování článků starších určité časové periody, hromadné odesílání emailů v určitý okamžik apod. Toto umožňuje unixový daemon[49] „Cron“, určený pro časově řízené automatické spouštění skriptů. Například v linuxových distribucích lze skripty webové aplikace nechat spouštět v pravidelných intervalech pomocí shellu[50], protože „Cron Daemon“ je nedílnou součástí Linuxu. Vývojáři projektu Habari se nechali inspirovat a vytvořili vlastní, jednoduchý Cron v PHP jazyce, založený na bázi unixového Cronu, který prozatím plní funkci mazání expirovaných záznamů o chodu systému (logů).

5.5.10 Vykonání požadavku

Na úplný závěr event handler[51] vykoná požadavek a veškerý uchovaný výstup skriptu v paměti serveru je odeslán klientovi, protože interpret narazí na funkci ob_flush(), která zadržený výstup z bufferu vyprázdní.

5.5.11 Controller

Controller projektu Habari pomocí vzoru Singleton[52] zabraňuje vytváření více než jedné instance třídy, aby nedocházelo k vykonávání určitých akcí, nebyly-li dosud ukončeny. Různé druhy požadavků, které uživatel vykonává na webových stránkách, vedou k různým akcím. Protože se tak děje z velké části přes klikání na odkazy, další podstatnou rolí Controlleru je tzv. Routing[53].

---

[49] *Program unixového původu, který mimo přímou uživatelskou kontrolu inicializuje procesy operačního systému.*
[50] *Software, poskytující v operačním systému uživatelské rozhraní.*
[51] *Objekt, který v aplikaci vykonává určité akce, jako reakci na událost v systému*
[52] *Singleton pattern – metoda, omezující vytváření instance třídy na maximálně jeden objektu*
[53] *Proces, při kterém se z URL získává název modulu, třídy, metody a parametrů a ukládá se do objektu požadavku.*

V případě redakčního systému Habari jde o způsob, kdy se pomocí několika tříd (rewriterules, URL) získají proměnné a jejich hodnoty z URL, na které jsou přesměrovány permalinky[54]. URL jsou parsovány[55] a hodnoty proměnných (přenášených v URL) jsou určujícími elementy ke zvolení Handlerů a akcí, které budou vykonávat.

5.6 Architektura redakčního systému Habari

Architektura systému je z velké části určena použitým návrhovým vzorem „Model-View-Controller“, jež definuje obecnou funkční logiku systému, manipulaci s daty a způsob prezentace vzhledu. Mnohem méně determinuje logiku a formu aplikačního rozhraní, jejíž vývoj a způsob řešení spočívá více na tvůrčích schopnostech vývojářů.

5.6.1 Princip modularity

V dnešní době si v oblasti programování upevňuje svoji pozici objektová orientace. Nástup objektově orientovaného programování způsobil lepší manipulaci s vytvořenými programy, jejich rozšiřování, aktualizaci a schopnost vzájemné kompatibility s jinými aplikacemi a moduly. Ten samý trend je možno zpozorovat i v oblasti redakčních systémů. Systémy postavené na jazyku PHP verze 4 v dobách, kdy ještě nebyl plně objektově orientovaný, začínají upadat a nastupuje vlna nových systémů, mnohem více otevřených a plně objektově orientovaných, jakým je i redakční systém Habari.

Tvůrci open source redakčních systémů dávají možnost prakticky komukoliv podílet se na jeho rozšiřování a právě to dělá ze systému univerzální nástroj, který tak lze využívat mnohem více způsoby. Aplikace by měla být rozčleněna do logických funkčních částí, se kterými lze pracovat jako s ucelenými jednotkami. Právě pro tuhle skutečnost se v programování používá pojem modularita, tedy že struktura systému dovoluje jednoduše měnit, přizpůsobovat, přidávat i odebírat jednotlivé součásti, a tak zabezpečit flexibilitu jeho nasazení v praxi.

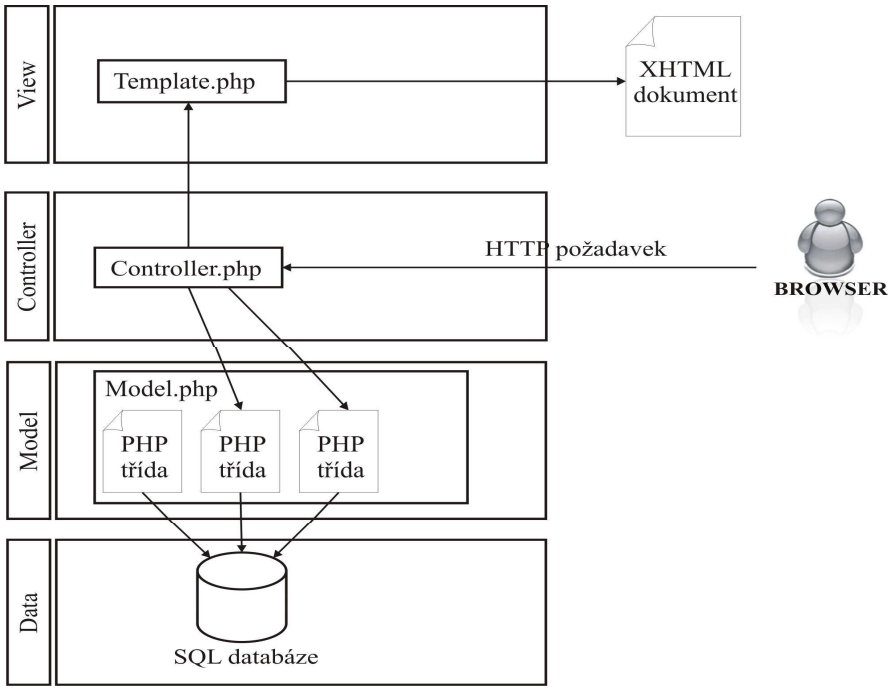
Modularita zároveň dovoluje programátorovi zapouzdřit jednotlivé funkční oblasti do modulů, které je možno využít v jiných oblastech v jiném kontextu. Právě modularita přispívá k přehlednosti struktury RS a dovoluje uživatelům zaměřit se jen na části, se kterými pracují bez znalosti jiných.

---

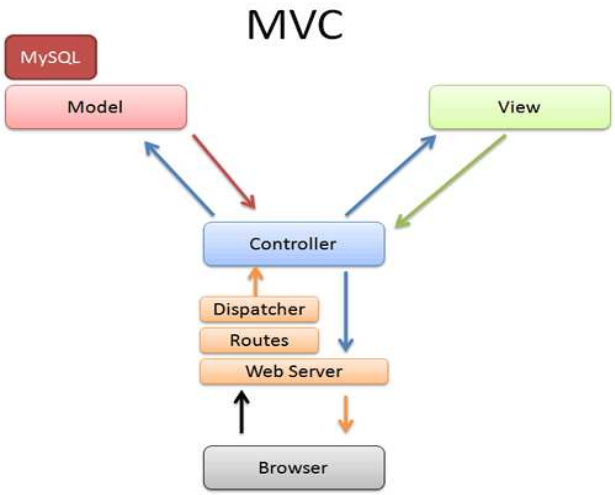
[54] *Permalink - esteticky upravená forma URL odkazů*
[55] *Parsing – proces identifikace a analýzy sekvence znaků k určení požadované struktury řetězce*

5.6.2 Struktura aplikace

Model-View-Controller je softwarová architektura, která rozděluje aplikaci na datový model, uživatelské rozhraní a řídící logiku. Komponenty jsou na sobě nezávislé a jejich modifikace mají minimální vliv na ostatní, mohou se tak vyvíjet nezávisle na sobě [22].



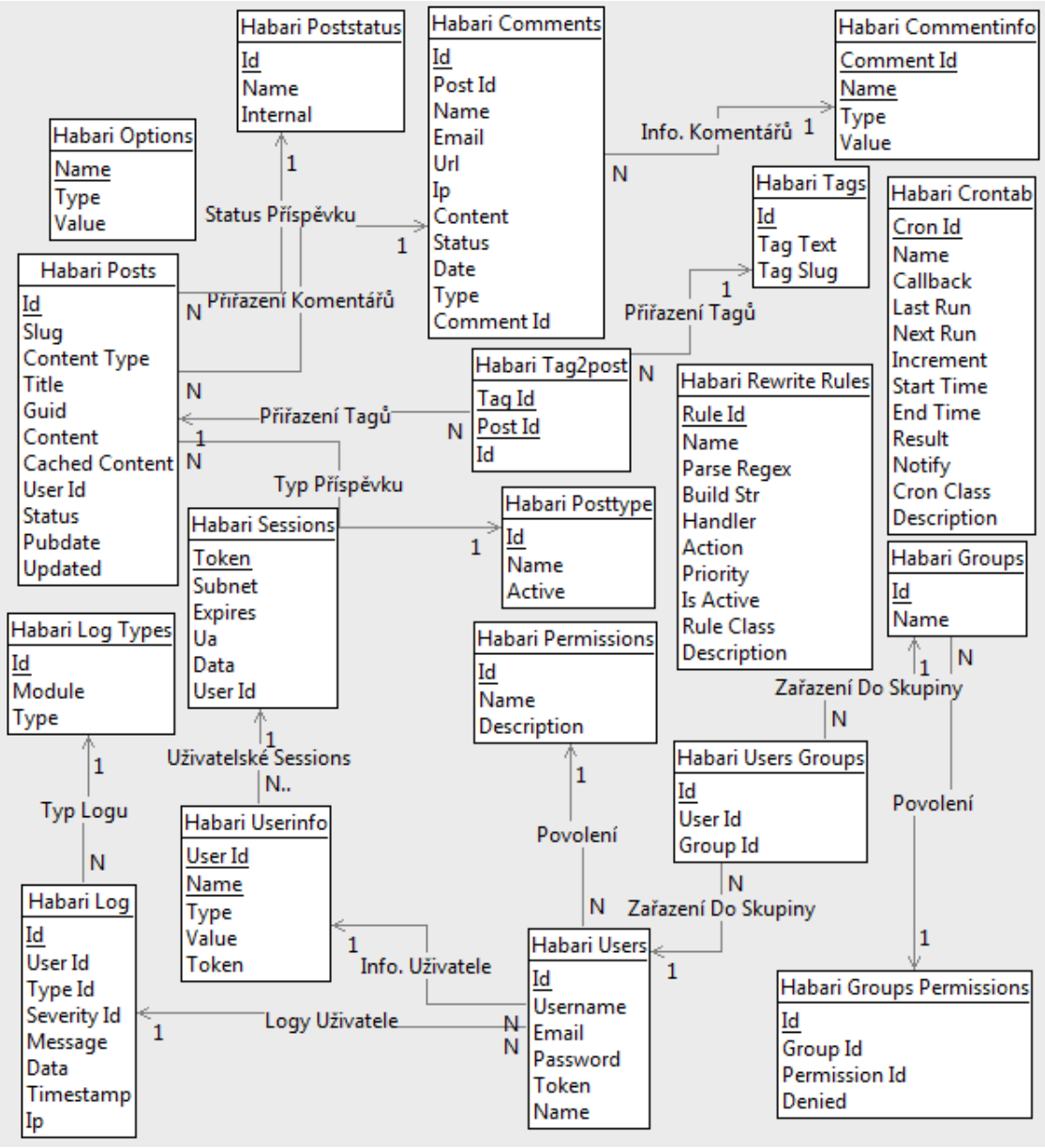
***Obrázek 3**: Další pohled na schéma Model-View-Controller (obrázek vytvořen v Eclipse)*



***Obrázek 4**: Schéma Modle-View-controller http://betterexplained.com/wp-content/uploads/rails/mvc-rails.png*

5.6.2.1 Model

Model tvoří datovou reprezentaci informací, jež aplikace využívá. Musí být nezávislý na ostatních částech a nesmí se na ně odkazovat, ale pouze poskytovat svoje metody částem View a Controller. To je důvod, proč může být jeden model velice dobře použit v různých aplikacích.



***Obrázek 5**: Schéma tabulek a jejich vzájemných vztahů v databázi (obrázek vytvořen v Eclipse)*

Definice tabulek databáze

- Users – Informace o uživatelích
- Userinfo – Uživatelská hodnost a čas autorizace, důležitý pro sessions[56]
- Sessions – Úložiště sessions
- Groups (vývoj) – Názvy uživatelských skupin
- User Groups (vývoj) – Pomocná tabulka vztahu M:N mezi uživateli a skupinami
- Group Permissions – (ve vývoji) – Pravomoci skupin
- Permissions – Pravomoci uživatele
- Log – Sklad veškerých záznamů o chodu systému
- Log Types – Typy záznamů
- Options – Sklad hodnot nastavení systému
- Rewrite Rules – Sklad nově definovaných událostí pro vývojáře
- Crontab – Sklad hodnot pro časově řízené události
- Post – Údaje o publikacích
- Posttype – Definované typy publikací
- Tags – Seznam použitých tagů[57]
- Tag2post – Pomocná tabulka vztahu M:N[58], kde publikace bývá označena více tagy, stejně jako tag může označovat více publikací
- Poststatus – Definované stavy publikací (publikované / rozepsané)
- Comments – Sklad komentářů
- Comments Info – Sklad informací o komentářích

5.6.2.2 View

View je komponentou, převádějící reprezentaci dat Modelem do formy vhodné k prezentaci uživateli. Prostřednictvím Controlleru má volný přístup k Modelu, díky metodám, které nabízí, ale nesmí měnit jeho stav.

Vzhled

Každý provozovatel website má odlišnou představu o jejím vzhledu, barevném sladění, fontech a celkovém rozložení prvků na stránce. To je jeden z důvodů, proč je design oddělen od obsahu, aplikační logiky i jiných částí. Díky tomu má uživatel možnost design stránek sám měnit, aniž by jakkoliv ovlivnil obsah nebo funkčnost. Uživatelské rozhraní se u redakčních

---

[56] Metoda zajišťující dočasnou, interaktivní výměnu informací mezi dvěma a více aplikacemi (browser – website), umožňující identifikaci uživatele za účelem zabránění opakujícím se autorizačním procedurám.
[57] Označení publikovaného obsahu za účelem kategorizace a třídění
[58] Vztah mezi 2 tabulkami, vyžadující jednu pomocnou k uložení všech možných kombinací, které mohou nastat

systémů v naprosté většině dělí na administrační a na rozhraní pro čtenáře website.

Šablony

Šablona zajišťuje vzhled prezentované informace. Stránky bývají v jednoduchém provedení složeny většinou z 5 šablon.

- Header (hlavička)
- Menu
- Body (hlavní část, tělo)
- Sidebar (postraní lišta)
- Footer (zápatí)

Šablonovací systém

Vývojáři publikačního systému Habari se rozhodli nevyužít prozatím žádného šablonovacího systému, ale prostého generování obsahu PHP skripty uvnitř HTML dokumentu. Ve vývoji je prozatím podpora šablonovacího systému Smarty a PHPTAL. Uživatelské prostředí je složeno z dílčích šablon. Je tomu tak, protože v publikačních systémech figuruje výrazné množství okolností, za kterých má být daný obsah zobrazen. Předmětem obsahu je sice publikace, článek nebo příspěvek, ale na websites jich je zpravidla množství, které vyžaduje transparentní přístup k obsahu pro pohodlí uživatelů.

Kdyby uživatelské prostředí tvořila pouze jedna šablona, tedy HTML dokument od hlavičky až po zápatí s vloženými PHP skripty, výsledkem by byla přílišná jednotvárnost a nutnost realizace složitých řešení. Proto je jednodušší vytvořit šablony specializované na zobrazování různého obsahu a volit taktiku přizpůsobení vzhledu obsahu, než obsah vzhledu.

Generování šablon v uživatelském prostředí

Handler pro uživatelské prostředí (userthemehandler) pomocí konstruktoru[59] získává informace o použitých šablonách a zajistí jejich zahrnutí do systému. Následně tento Handler zpracuje požadavek uživatele a zobrazí šablonu, příslušící danému obsahu. Uživatelé při pohybu na websites mají možnost zobrazovat pouze jeden konkrétní článek, sérii článků v časovém sledu, články určitého označení, kategorie, měsíce či roku publikace, výsledky vyhledávání a další. To jsou požadavky, kterých se ujímá právě „userthemehandler“.

Dále existují šablony, které nejsou volány Handlerem na zobrazení podle druhu požadavku. Jsou to šablony, které reprezentují část uživatelského prostředí, zprostředkovávající určitou funkčnost uživateli. Může to být celá

[59] *__construct( ) - Speciální metoda jazyka PHP – třídy obsahující konstruktor, ho volají na každý nově vytvořený objekt. Jedná se o metodu, která se vykoná při vytváření každé instance třídy (operace prvotního nastavení třídy)*

postranní lišta, která poskytne prostor pro kategorizaci příspěvků, aktuální novinky nebo prostor pro výstup pluginů. Tyto šablony mohou být v případě např. postranní lišty, záhlaví nebo zápatí volány při každém uživatelově požadavku.

Generování šablon v administračním prostředí

Šablony, kterých využívá administrační prostředí, jsou podle požadavků generovány handlerem, vytvořeným speciálně pro tuto oblast. Administrační prostředí má k dispozici odlišné šablony a změnit jeho vzhled je mnohem komplikovanější. Už jen proto, že vzhled a funkčnost tohoto prostředí je z velké části zajištěna pomocí Ajaxu a Javascriptu.

Šablony pro výstup pluginů

Výstup z pluginů do uživatelského prostředí hraje důležitou roli. Implementace pluginů do systému je v projektu Habari provedena velice jednoduše a výstup pluginu rovněž. Nejjednodušším způsobem pro tvůrce pluginů je vytvořit vlastní šablonu ve formě souboru s „ “[60] odstavcem, ve kterém je PHP skripty generován obsah, respektive výstup pluginu. Pomocí jednoduchého HTML odstavce tvůrce pluginu určí své místo CSS stylováním a zahrnutím (include) šablony mezi ostatní.

5.6.2.3 Controller

Controller je řídící komponentou. Reaguje na uživatelské požadavky a události a zajišťuje vykonání změn v Modelu a View.

Web server přesměruje uživatelský požadavek (ve většině případů webových aplikací na „index.php“), z URL se získá název modulu, třídy, metody a argumentů (routing), a vše je uloženo do objektu požadavku. Tento objekt je předán všem částem Controlleru, které si o něj zažádají. Dispatching[61] pak zajistí vytvoření instancí Controlleru a volání konkrétních metod.

V projektu Habari probíhá routing metodou „parse_request()“ a dispatching metodou „dispatch_request()“.

Controller.php

```
<?php Controller::parse_request();?>
<?php Controller::dispatch_request();?>
```

Zdrojový kód 1: Dvě nejdůležitější funkce v souboru Controller.php

[60] *HTML element, který definuje sekci a přiřadí jí pozici v HTML dokumentu*
[61] *Proces, při kterém se vytváří instance controlleru a volá se jeho konkrétní metoda.*

5.6.3 Aplikační rozhraní systému (API)

Aplikační rozhraní, jeho důmyslnost a logika, je jedním z nejdůležitějších elementů především u open source aplikací. Představuje komunikační vrstvu mezi jádrem a doplňkovou nadstavbou systému. Díky její existenci nemusí tvůrci rozšíření nutně znát logiku a architekturu aplikace a právě v open source projektech, kde se na jejich vývoji podílí kdokoliv, je jednoduchost a efektivnost řešení určujícím prvkem narůstajícího počtu rozšíření.

5.6.3.1 Event handlers

Úkolem Handlerů je zhodnocení zpracovaných požadavků a vykonání odpovídající reakce. V případě redakčních systémů jde hlavně o vygenerování obsahu uživatelského rozhraní. Obecně je v publikačních systémech používáno mnoha typů Handlerů, ale všechny pracují na stejném principu. Jeden z typů slouží publikační části pro řadové návštěvníky website (userthemehandler), další administračnímu prostředí (adminhandler), procesu instalace, autorizace, RSS zdrojům, ale i zpracování Javascriptových a Ajaxových požadavků. Tato prostředí se většinou diametrálně liší, protože jsou vytvořeny pro odlišné uživatelské potřeby. Navíc jednotlivé metody různých tříd potřebují vzájemně komunikovat.

Handler je třída, která musí v publikačním systému Habari zajistit diverzifikaci, ale zároveň i unifikaci různých druhů metod a událostí podle funkce v systému. Tuto potřebu podmiňuje také skutečnost, že handlerů je 7 jen v jádře systému, každý pro jednu specifickou oblast.

ActionHandler.php

```
   /* Všechny handlery musí implementovat funkci act(), aby se přizpůsobily danému
   API. Tato je defaultní funkcí, volající metodu „act_$action()“ instance třídy
   „actionhadler“. Přičemž ostatní „handler classes“ tuto rozšiřují a funkci „act()“
   případně přepisují vlastní funkcí “act()“. Tato funkce není pouze pokynem pro
   vykonání reakce na událost, ale i signálem pro pluginy, které mají reagovat před
   nebo po hlavní události. Proto jsou v handlerech implementovány prefixy různého
   druhu. /*

   public function act($action) {
     $this->action= $action;

     $action_method= 'act_' . $action;
     $before_action_method= 'before_' . $action_method;
     $after_action_method= 'after_' . $action_method;
     if (method_exists($this, $action_method)) {
               if (method_exists($this, $before_action_method)) {
                         $this->$before_action_method();
               }
               Plugins::act( $before_action_method );
               $this->$action_method();
               Plugins::act( $after_action_method );
               if (method_exists($this, $after_action_method)) {
                         $this->$after_action_method();
               }
       }
}
```

Zdrojový kód 2: Funkce act() v souboru ActionHandler.php

5.6.3.2 Technické provedení aplikačního rozhraní

K zajištění výše uvedených specifik byla vytvořena metodika, která představuje jakési API mezi jednotlivými funkcemi v systému a handlerem. API určuje, jak má být nakládáno s metodami, které dodržují ve svém názvu syntaxi, definovanou aplikačním rozhraním, respektive jakou syntaxi mají dodržovat například tvůrci pluginů nebo šablon. Jedná se přitom o metody, které jsou specifické svým posláním vykonávat jistou akci, která je v systému běžná, často a na různých místech vyžadovaná.

```
Controller.php

public static function dispatch_request() {

Plugins::act('handler_' . Controller::instance()->action,
Controller::get_handler_vars());

     /* Funkce act() třídy Plugins, je v podstatě handlerem událostí, určených ke
zpracování pluginy. Bude-li někdo chtít vytvořit plugin, který se bude
inicializovat, respektive plnit svoji funkci, při každém požadavku, pak callback[62]
tohoto pluginu musí mít tvar, odpovídající právě této pasti, respektive předřazen
string „handler_“ */

if(method_exists(Controller::instance()->handler, 'act')) {
     Controller::instance()->handler->act(Controller::instance()->action);
               }
     /* Účelem této metody je přiřadit funkci act() odpovídající instanci právě toho
handleru, který má tento typ akce vykonat. Funkcí act() je mnoho a přiřazením
instance odpovídající třídy, resp. Handleru, se vybere ta pravá. Tato určení
proběhnou na základě získání informací Routingem URL (např. handler =
userThemeHandler, action = display_post) */

   }
```

Zdrojový kód 3: Detail funkce dispatch_request() v souboru Controller.php

5.6.3.3 Princip komunikace pluginů s aplikačním rozhraním

Pluginy jsou založeny na systému tzv. akcí a filtrů. Nastane-li v redakčním systému nějaká událost (uložení, zobrazení příspěvku atd.), tak díky definované „pasti“ („plugins::act()“) na tuto událost dojde k inicializování pluginu a k předání argumentů, aby mohl na událost reagovat. Přičemž předané hodnoty zůstávají konstantní. Filtry fungují podobně, ale jejich účelem není inicializace pluginu k vykonání nějaké reakce na událost, ale předání hodnot za účelem jejich vyhodnocení, modifikace a navrácení.

Vývojář pak díky funkci, jejímž výstupem je požadovaný efekt pluginu, může jejím použitím jako callback v nalíčené pasti tento výstup dostat kam potřebuje. Komunita kolem projektu má tak možnost vytvářet rozšíření, aniž by měla znalosti o principu fungování systému.

---

[62] *Funkce, kterou přijímají určité funkce jako argument, např. “call_user_func_array( )“ – jejím výstupem pak bude výstup callbacku uvnitř*

```
Plugins.php

   /* Toto je funkce zmiňované pasti, respektive funkce, kde metoda
   „call_user_func_array()“, vrací výstup funkce (tzv. callback), kterou má v sobě
   jako argument. Tento callback je metodou pluginu, takže její výstup, respektive
   výstup pluginu, se tímto zobrazí v místě naličené „pasti“. Systém dokáže funkci v
   aplikaci bezpečně najít a vykonat, protože argument (typu pole), se skládá z
   funkce i jména její třídy, případně instance objektu, které slouží jako
   identifikátory. Na tomto principu nefungují jen pluginy, ale částečně i výkonná
   vrstva redakčního systému. /*

public static function act()
{
   $args = func_get_args();
   $hookname = array_shift($args);
   if ( ! isset( self::$hooks['action'][$hookname] ) ) {
     return false;
   }
   foreach ( self::$hooks['action'][$hookname] as $priority ) {
     foreach ( $priority as $action ) {

               call_user_func_array( $action, $args );
               }
     }
   }
```

Zdrojový kód 4: Detail funkce act() v souboru Plugins.php

5.7 Rozšíření

Plugin neboli rozšíření jsou skripty, přinášející do aplikace novou funkčnost. V open source publikačních systémech hrají rozšíření jednu z nejdůležitějších rolí. Jsou totiž vytvářeny komunitou kolem projektu a zajišťují jeho růst a použitelnost. Již ve fázi vývoje jádra aplikace musí mít vývojáři promyšlenou metodiku komunikace pluginů s aplikačním rozhraním. Této metodiky se potom musí pluginy vytvářející komunita řídit.

Z této skutečnosti vyplývá, že jedním z určujících faktorů růstu komunity kolem projektu je právě originálnost a transparentnost standardu, respektive doporučeného způsobu implementace rozšíření a snadnost napojení na API. Není-li dostatečně efektivní, tvorba pluginů se začne vyhýbat standardním postupům a systém se stane více nepřehledným a problémovým. Aplikace většího množství pluginů pak vede k častým chybám. Jednotlivé procesy trvají déle, případně vlákna zanikají v nelogických smyčkách. Rozšiřování funkčnosti pomocí pluginů tedy nesmí jít na úkor svižnosti aplikace.

Rozšíření se typově liší, může se jednat o jeden soubor kódu, ale i desítky souborů s desítkami nových tříd. V takovém případě často probíhá instalace[63] rozšíření, případně dochází v aplikaci k úpravě vývojářem nebo uživatelem. To se ale děje zřídka a v případě projektu Habari čekají masivnější rozšíření ještě na své tvůrce a to včetně systémových pluginů, respektive rozšíření v základním instalačním balíčku.

[63] *Automatická úprava kódu na straně rozšiřované aplikace*

5.7.1 Umístění pluginů

Jednoduchý plugin je soubor ve tvaru „název.plugin.php“, umístěný v adresáři, ve kterém bude systémem rozpoznán. Všechny pluginy musí rozšiřovat třídu „Plugin“, která definuje metody pro nalezení a registraci pluginu. Funkce pluginu mohou být volány buď při konkrétním, či každém požadavku.

5.7.2 Podcasting

Nejdůležitějšími pluginy jsou Simplefilesilo, Flickrsilo a Viddlersilo. Tato rozšíření mají v redakčním systému Habari na starosti tzv. podcasting[64]. Publikace a správa souborů, obrázků a videí je díky těmto rozšířením na vysoké úrovni. Vývojáři se zaměřili na transparentnost a jistou míru unifikace, nabízející uživatelům přehled a lehkost ovládání, při vysoké efektivitě.

Tvůrci pluginů se rozhodli spolupracovat s webovými aplikacemi „viddler.com“ a „flickr.com“, umožňujícími uživatelům nahrávat videa a obrázky na server a sdílet je s ostatními. Obě tyto aplikace mají zveřejněny klíčové prvky svého API, což nabízí možnost přístupu a manipulace s daty na „Viddler serveru[65]“ prostřednictvím jiných webových aplikací, jako je Habari. Je přitom nutno zažádat si o povolení pro vykonávání této činnosti a splnit jisté podmínky (např. maximálně 1 požadavek na přehrání videa za sekundu) [23].

5.7.2.1 Přínos pro redakční systém

Vlastní-li uživatel webovou kameru, stačí několik kliků myší v administračním prostředí k nahrání videa, jeho uploadu na „Viddler server“, případně rovnou k jeho publikaci. Přidá-li se k tomu přehledná správa všech uživatelských videí přímo v administraci, která má díky technologii Ajax zároveň stylové uživatelské prostředí, tak je i po grafické stránce vše slazeno se vzhledem vydařeného flash přehrávače, který lze prostřednictvím webové aplikace Viddler využívat přímo na stránkách redakčního systému habari.

Podobně je to i s rozšířením Flickrsilo, ale elementem jsou v tomto případě obrázky. SimplefileSilo slouží převážně ke správě souborů na serveru a zpřístupnění souborů ke stažení z webových stránek, což lze velice jednoduše díky automatické tvorbě html odkazu na daný soubor.

5.7.2.2 Viddlersilo Plugin

Plugin viddlersilo je tvořen soubory „viddlersilo.plugin.php“ a „vidfunc.js“. Obecný princip jeho funkce je vygenerování požadavku na API aplikace Viddler přes http protokol metodou GET nebo POST a získání odezvy.

---

[64] *Metoda šíření nejčastěji zvukových nebo video záznamů*
[65] *Viddler server – označení pro server, na kterém běží aplikace viddler*

Co se požadavku týče, je zaměřen na dvě oblasti: získávání informací (uživatelské údaje, ID, informace o účtu, detaily o videu apd.) a upload videa.

Curl

Pro jazyk PHP je k dispozici velice užitečná knihovna „libcurl“, která podobně jako jednotlivé moduly v archivu CPAN[66] u jazyka Perl, obohacuje PHP interpret o novou funkčnost a třídy, zaměřené na transparentní přenos dat pomocí nejrůznějších protokolů, ale hlavně pomocí protokolu HTTP. Curl byl původně vytvořen jako PHP nástroj pro „command line interpreter“[67], takže lze jeho předností využívat mnoha způsoby [24].

Komunikace s aplikačním rozhraním vzdáleného serveru

Pro získávání informací přes API vzdálené aplikace stačí vygenerovat potřebnou metodu, definovanou aplikačním rozhraním Viddleru (např: videos.upload, videos.getDetails, users.getProfile, users.register) s potřebnými argumenty a funkcí „curl_setopt()“ vytvořit HTTP požadavek na „Viddler server“. Odpovědí jsou vygenerovaná data, zapouzdřená pomocí jazyka XML do hodnot, které jsou parserem po přijetí transformovány do pole. Tyto informace vyžaduje Viddlersilo plugin pro další vzájemnou komunikaci, především pro upload videí.

Upload videí

Pro upload libovolných nebo přímo v administračním prostředí nahraných videí je situace následující. Tento úkon v podstatě navazuje na předchozí fázi získání informací. Aby mohlo být video na „Viddler server“ nahráno, potřebuje funkce curl_setopt obdržet cílovou destinaci uploadu videa a další informace. Je-li video umístěno na „Viddler serveru“, pak je veškerá komunikace mezi redakčním systémem a aplikací Viddler při přehrávání videí uživatelem, zprostředkována flash přehrávačem. Meta-informace o videu se přitom ukládají do cache[68] souborů aplikace Habari.

Stavba pluginu

Konkrétně plugin Viddlersilo je z velké části předepsaný aplikací Viddler tak, aby mohl komunikovat s jejím API. Respektive jedna z jeho dvou tříd je až na drobné úpravy téměř identická s poskytnutou. Druhá třída už je zkonstruována typickou metodikou pro komunikaci pluginu s domácím aplikačním rozhraním.

---

[66] *Comprehensive Perl Archive Network – archiv více než 12 000 modulů a knihoven, ale i dokumentace jazyka Perl*
[67] *CLI – program operačního systému, který interpretuje uživatelem zadaný vstup*
[68] *Dočasné úložiště frekventovaně využívaných dat*

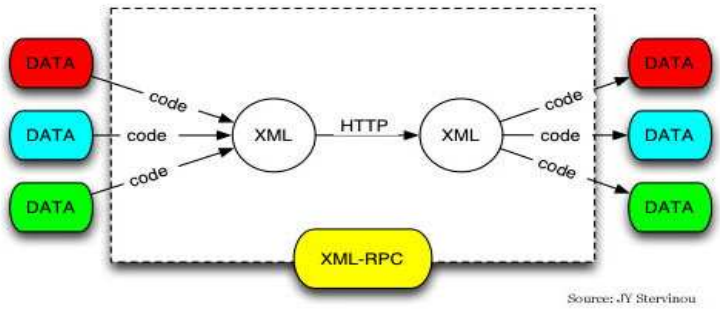
5.7.3 Pingback

Pingback je obecně známá metoda pro publicisty na internetu, jak být informován o zpětném odkazu na jejich článek, respektive jak informovat jiné publicisty, na které se ve svých publikacích odvolávají odkazem. Užitek tato metoda přináší i čtenářům příspěvku, kteří si mohou přečíst i publikaci autora na příspěvek odkazujícího, která bude s největší pravděpodobností podobné tématiky. Několik řádek kolem odkazu v odkazující publikaci se zobrazuje pod článkem, v sekci komentářů a reakcí čtenářů. Čtenáři pak mají možnost projít si všechny publikace odkazující na článek, které ovšem musí podporovat metodologii Pingback [25].

5.7.3.1 Forma přenosu dat

Proveditelnost vyžaduje sestavení HTTP hlavičky s potřebnými údaji (typ požadavku: POST / GET, Host[69], User-agent[70], apd.) a zapouzdření obsahu do XML formátu. Tato forma je považována za standardní metodiku a využívá specifikace XML-RPC[71]. Podle názvu je patrné, že umožňuje vzdálené volání procedur za využití protokolu HTTP pro přenos dat a specifikace XML pro jejich kódování. XML-RPC je v případě publikačních systémů řešeno architekturou klient/server.

Princip spočívá ve vytvoření klientského požadavku metodou POST, sestávajícího s HTTP hlavičky a vygenerovaného názvu procedury, kterou voláme, a pro nás klíčových parametrů. Obsah požadavku je zapouzdřen podle xml specifikace a odeslán spolu s hlavičkou, kde je definována adresa xml-rpc serveru. Adresa xml-rpc serveru může být případně přímo v URI. Na straně serveru je požadavek přijat, obsah dekódován a dojde k inicializaci volané metody za použití doručených argumentů. Odpověď klientovi přijde opačným způsobem, přičemž zapouzdřený obsah nese místo informací o volané metodě elementy, obsahující námi požadovaná data [26].



Obrázek 6: *Schéma principu specifikace XML-RPC*
http://static.userland.com/images/XML-RPC/xmlrpc.jpg

[69] *Atribut s hodnotou doménové adresy serveru*
[70] *Atribut s hodnotou názvu browseru, operačního systému, kódování apd.*
[71] *Extensible Markup Language - Remote Procedure Calling*

5.7.3.2 Princip metody pingback

Předešlý odstavec dokumentuje formu provedení samotného procesu komunikace a přenosu dat mezi servery, na které je postavena metoda Pingback. Tato metoda je opět založena na klient/server architektuře, ale už neřeší komunikaci mezi servery, ale rozpoznávání Pingback serverů. Metoda Pingback totiž spočívá v označení html dokumentů, aby mohly být zapojeny do hry. Klient sehrává svoji roli v případě publikace článku A, odkazujícího na článek B. Prozkoumá, jestli je hlavička dokumentu označena vlastností „X-pingback" s hodnotou adresy xml-rpc serveru. Jestli-ano, odešle na xml-rpc server požadavek se dvěma atributy (URL článku A a B). Server klientský požadavek prověří a sám vytvoří xml-rpc požadavek na klienta, aby získal konkrétní data ohledně článku A.

Tento proces v projektu Habari probíhá ve chvíli publikování článku a jeho ukládání do databáze, případně editaci článku. Dojde ke skenování všech odkazů ve článku a k uvědomění autora článku B. V pluginu je dále řešeno automatické obohacení hlaviček příspěvků o vlastnost „X-pingback“ s adresou na server. Samozřejmostí je i přijímání xml-rpc požadavků klientských aplikací xml-rpc, kde je naopak odkazováno článkem B na článek A.

```
<?php if ( $post->comments->pingbacks->count ) : ?>
  <h1>
  <?php
   echo $post->comments->pingbacks->count . ' ';
   echo _n( 'Pingback', 'Pingbacks', $post->comments->pingbacks->count );
   echo ' ' . to ' ' . $post->title; ?>
 </h1>
  <div id="pings">
   <ol id="pings-list">
    <?php foreach ( $post->comments->pingbacks->approved as $pingback ) : ?>
       <li id="ping-<?php echo $pingback->id; ?>">
         <p>
           <?php
echo "<a href=\"{$pingback->url}\">{$pingback->name}</a> &raquo; {$pingback->content}";
           ?>
         </p>
       </li>
    <?php endforeach; ?>
   </ol>
 </div>
<?php endif; ?>

       /* Skript zjistí, zdali existuje schválený pingback pro zobrazený článek,
případně vypíše počet a jsou-li nějaké schváleny administrátorem, zobrazí URL
odkazující stránky, titulek a v defaultním nastavení 5 řádek kolem odkazu. /*
```

Zdrojový kód 5: Příklad, jak by mohl vypadat kód na zobrazování pingback informací

5.7.3.3 Pingback v systémových záznamech

V administračním prostředí publikačního systému Habari se nachází přehled nejrůznějších záznamů, logů[72], kde se zaznamenávají události, které

[72] *Záznam chronologicky ukládaných dat pro pozdější použití*

v systému nastaly. Úkolem třídy „logentry“ je skladování dat o nastalých událostech do databáze, třída „eventlog“ zajišťuje jejich filtraci a výběr podle přání uživatele. Zobrazení a rozhraní pro uživatele je vyřešeno velice prakticky s možností vyhledávání logů v databázi. Podle šesti kritérií (časové období, uživatel, modul, typ a druh záznamu, IP adresa uživatele) má administrátor snadnou cestu k objevení požadovaného záznamu. Zadá-li v kritériích druh záznamu „pingback“, zobrazí se mu údaje o všech příchozích, odchozích i neúspěšných pokusech o pingback spolu s údaji o odkazujícím článku a jeho adrese.

5.8 Nároky na neformální členství v komunitě projektu Habari

Komunita ve světě informačních technologií a internetu je založena na stejných hodnotách a principech jako komunity v reálném světě. Členy spojují stejné hodnoty, záliby nebo styl života. Členy je dělá i skutečnost, že dodržují normy a splňují očekávání dané komunity, protože v některých, jako je Habari, je potřeba dovedností a zkušeností nutných pro členství z důvodu charakteru jádra, respektive středu zájmu komunity, má-li nějaký.

Představený project management, jehož části a využívané technologie se měnily v čase, umožňuje sociální kontakt mezi členy. Jedna z nejdůležitějších podmínek dlouhotrvající existence komunity kolem open source projektu. Nelze se podílet na vývoji aplikace bez vzájemné komunikace pracovního rázu, ale ani bez sociálního kontaktu s ostatními, protože to je jedna z podmínek, která komunitu spojuje a drží členy zainteresované, nadšené a odhodlané. Tyto skutečnosti odpovídají spíše nárokům na vynaložený čas.

Nároky na člena komunity mohou být ale i dovednostní. I v pouhé komunikaci, kdy v komunitě projektu Habari vládne jednotný jazyk, angličtina, přestože většina členů nejsou rodilí mluvčí. Veškeré zmíněné aplikace project managementu (Subversion, Trac, PhpDocumentor) vyžadují praxi a předchozí zkušenosti k jejich plnohodnotnému využívání ve prospěch projektu. V neposlední řadě, tím nejdůležitějším požadavkem je obecná znalost programování.

Podílet se na vývoji jádra systému vyžaduje nejenom znalosti předchozí, ale i ponětí o základních návrhových vzorech používaných ve webových aplikacích a frameworcích, znalost databáze a serveru, kreativní myšlení a trpělivost při nekonečném výskytu chyb a nalézání řešení pro jejich nápravu.

Na vývoj rozšíření a vzhledu jsou kladeny mnohem mírnější požadavky. Po nastudování funkcí aplikačního rozhraní a jejich účelu nepotřebuje tvůrce o redakčním systému mnoho vědět. Pro tvorbu vzhledu je ale třeba designérských zkušeností grafika.

5.9 Zhodnocení redakčního systému Habari

Projekt Habari je svojí fází vývoje redakčním systémem, vhodným spíše pro zkušenější uživatele, kteří ho použijí pro tvorbu svého website jako framework. Tito uživatelé se většinou na oplátku za jeho použití podílejí s ostatními o rozšíření, která vyvinuli, čímž zajistí projektu jeho budoucí plnohodnotnou použitelnost i pro běžné uživatele.

Filosofií projektu je využívání řešení, která nebudou překážkou v budoucím vývoji systému, a která setrvají aktuální v dlouhodobém průběhu času. Projekt je v rukou vývojářů, kteří se od prvopočátků z velké části podíleli na tvorbě podobného projektu, poučili se z chyb a mají tedy patřičné zkušenosti a předpoklady, aby vytvořili redakční systém, který možná bude v budoucnu pohánět milióny webů. Tito vývojáři na něm bez sebemenší ztráty zájmu denně pracují a snaží se ho mediálně prosadit. Příkladem může být registrace projektu Habari na letošní sezónu Google Summer of Code a získat větší popularitu a stipendia pro vývojáře, kteří by na redakčním systému pracovali. Projekt sice vybrán nebyl, ale v takové konkurenci jako na GSoC to není překvapující.

5.9.1 Srovnání redakčního systému Habari s ostatními

Projekt se nachází na začátku fáze, ve které se rozhodne o jeho budoucí úspěšnosti. Po dokončení jádra, které není nikterak zajímavé pro obyčejné uživatele bez zájmu o vývoj, se začínají vytvářet doplňky a rozšíření, které běžný uživatel teprve dokáže ve spojení s jádrem ocenit. Uživatelská oblíbenost redakční systém zviditelní, a tak dostane možnost zaujmout vývojáře, kteří se ho sami rozhodnou používat a ve výsledku vylepšovat o rozšíření, o které jsou ochotni se s ostatními podělit.

Zmíněný sled událostí je řetězovou reakcí, kterou vyprovokuje dobré jádro a zdravá komunita ochotná systém mediálně zviditelnit a poskytnout uživateli možnost jej ocenit.

5.9.1.1 Z hlediska architektury a technologie

Na úvod je potřeba říci, že Habari je na poli redakčních systémů v jazyce PHP jedním z nejnovějších projektů. Na internetu lze samozřejmě nalézt velké množství projektů novějších, ale většinu z nich nelze považovat za čistokrevný redakční systém, protože jsou postaveny na frameworku (Symphony, Zend Framework) nebo jsou to mnohdy „rychloprojekty“, vypadajícími na první pohled plnohodnotně. Postrádají za sebou totiž velkou spoustu práce, která je dělá perspektivními do budoucna.

Vyskytují se i takové redakční systémy, které jsou podle tvůrců vytvořeny jazykem PHP verze 5, a současně uvádí, že minimální systémový požadavek na interpret jazyka je PHP verze 4, což si protiřečí. Každopádně většina nejznámějších redakčních systémů díky použití jazyka PHP 4 není plně

objektově orientována. Následky se neprojevují v rychlosti systému, ale spíše v otázce budoucího vývoje.

5.9.1.2 Z hlediska funkčnosti

Vzhledem k roku a půl existence projektu se Habari nemůže rovnat ostatním publikačním systémům jako je Wordpress, Joomla nebo Drupal. Proto je tento systém prozatím vhodnější spíše uživatelům, upřednostňujícím vlastnoruční vytvoření funkčnosti, kterou potřebují. Díky flexibilnímu API lze použít s malými úpravami pluginy, vytvořené pro jiné redakční systémy. Rozšíření jsou nedostatečná jak po stránce kvalitativní, tak i kvantitativní. Proto může Habari na řadového uživatele působit špatným dojmem a vzbuzovat obavu neposkytnutí dostatečných možností při jeho použití.

Co se týče jádra systému, tak v porovnání s konkurencí je na vysoké úrovni, hlavně po stránce funkční logiky. Vývojáři zvolili cestu „dvakrát měř, jednou řeš“ na úkor pomalejšího tempa vývoje. Nevýhodou je dosavadní absence uživatelských skupin a rozdělení podle výše privilegií v administračním prostředí, což je poslední nehotová část jádra.

5.9.1.3 Z hlediska potenciálu komunity

Jádro komunity projektu Habari je založeno na přátelských vztazích, dobré morálce a častých osobních setkáních členů. Silnou stránkou je její dobrá morálka vyplývající ze zkušeností zakladatelů, kteří zažili mnoho pádů a vzestupů projektů a jejich komunit. Komunita je ovšem zatím nedostatečně zajištěná, co se kvantity týče. Oproti stovkám stálejších členů konkurenčních redakčních systémů má projekt Habari pouhé desítky.

Po technologické stránce komunita využívá těch nejlepších dostupných prostředků správy webového projektu, které představují pohodlné prostředí všem vývojářům. Jelikož kvalita project managementu je jedním z několika důležitých faktorů, podle kterých si vývojář projekt vybírá, mohla by tato skutečnost být jednou z klíčových pro růst redakčního systému Habari.

5.9.2 Obecné závěry

Na základě zjištěných faktů je perspektiva redakčního systému Habari jednak v rukou jeho zakladatelů, kterým se musí podařit projekt medializovat a zviditelnit, aby vůbec mohl být oceněn. V rukou celé komunity je nejen medializace projektu, ale hlavně tvorba doplňků a vzhledů, aby tak byl využit vynikající potenciál samotného jádra systému, které dělá projekt Habari zajímavým a velice konkurenceschopným. Množství doplňků a druhů vzhledu je snad nejdůležitějším faktorem při rozhodování uživatele o volbě redakčního systému. Podaří-li se komunitě úspěšně zvládnout tuto fázi vývoje, projektu Habari by nemělo stát v cestě na vrchol mnoho překážek.

6 Závěr

Lidé jsou v dnešní době závislí na informacích, a jelikož internet je jejich největším zdrojem, stává se čím dál více součástí lidských životů. Poznání redakčních systémů by jedinci výrazně pomohlo pochopit jeho principy a vytvořit si o trochu přesnější obraz o prostředí, ve kterém se více či méně pohybuje.

I přes velký rozmach redakčních systémů v posledních letech je jejich důležitost podceňována. Mnoho lidí a firem má potřebu vytvořit si vlastní webové stránky, ale jejich nedostatečné znalosti této problematiky jim to nedovolují a navíc o existenci redakčních systémů nemají vůbec ponětí. Protože není dostatek takových zdrojů informací, které tyto sféry lidí oslovují, převládá tendence nechat si zhotovovat websites na zakázku. V České republice je tato oblast trhu velice vyspělá, s vysokou a kvalitní nabídkou těchto služeb, což dokumentuje i vysoká úroveň českého internetu. To ovšem nelze říci o převážném zbytku Evropy, kde jsou na tvorbu websites vynakládány veliké finanční výdaje a kde by větší povědomí o redakčních systémech kvalitě a růstu internetu a hlavně lidem a firmám velice pomohlo.

Jedním z cílů práce bylo zmapování situace v oblasti publikačních systémů, což bylo provedeno jak představením jejich možného rozdělení podle specializace obecně, tak i pohledem konkrétním na jedny z nejznámějších publikačních systémů a jejich budoucí perspektivu. Výsledky hodnocení uvedených systémů korelují se sílou komunity kolem nich, která jim garantuje udržení se na předních pozicích v oblíbenosti nebo pozvolný zánik.

Praktická část práce odhalila svým technickým zaměřením architekturu, principy funkce a filosofii publikačního systému Habari, čímž umožnila prozkoumat slabé i silné stránky systému, určit fázi v časovém kontinuu vývoje, ve které se nachází a definovat důvody a podmínky jeho možného budoucího umístění mezi nejlepšími. Nejvýznamnějším důvodem je nepochybně kvalitní jádro systému a potenciál komunity projektu. Její průřez odhalil zákulisí a principy vývoje projektu Habari, který je závislý na síle, odhodlání a schopnosti zakladatelů a komunity medializovat redakční systém Habari, aby mohl být uživateli vůbec oceněn a vynesen na výsluní.

7 Seznam literatury

[1] GUTMANS, Andi. *Mistrovství v PHP*. Praha: Computer Press 2005. ISBN: 978-80-251-1519-0

[2] April 2008 Web Server Survey, http://www.netcraft.com (online, 20. 4. 2008)

[3] GERNER J., NARAMORE E., OWENS M.L, WARDEN M. *Professional LAMP: Linux, Apache, MySQL, and PHP5*. Indianapolis: Wiley Publishing 2006. ISBN-10: 0-7645-9723-X

[4] HINZ S., DUBOIS P., STEPHENS J. *MySQL administrator`s guide and language reference*. USA: MySQL Press 2006. ISBN: 0-672-32870-4

[5] GNU Genereral Public License, http://www.gnu.org/licenses/gpl.html (online, 27. 3. 2008)

[6] GNU Lesser Genereral Publi License, http://www.gnu.org/licenses/lgpl.html (online, 27. 3. 2008)

[7] Licences, http://www.fsf.org/licensing/licenses/index_html (online, 27. 2. 2008)

[8] Moodle statistics, http://moodle.org/stats (online, 17. 4. 2008)

[9] Development Working Group, http://dev.joomla.org/content/view/1992/53 (online, 13. 2. 2008)

[10] GRAF H.; *Building Websites With Joomla!*. Birmingham: Packt Publishing Lts. 2006. ISBN 1-904811-94-9

[11] Google to invest 105,000 USD in Drupal, http://buytaert.net/google-to-invest-105000-usd-in-drupal (online, 23. 4. 2008)

[12] DOUGLASS R., LITTLE M., SMITH J.; *Building Online Communities with Drupal, phpBB and Wordpress*. New York: Apress 2006. ISBN: 1-59059-562-9

[13] PATERSON, Douglas. *Building Websites With PHP-Nuke.* Birmingham: Packt Publishing Lts. 2005. ISBN 13 978-1-904811-05-3

[14] Typo3 Statistics, http://www.webformat.com/en/typo3 (online, 13. 4. 2008)

[15] Total Members, http://www.php-fusion.co.uk/news.php (online, 13. 4. 2008)

[16] FAQ, http://wiki.habariproject.org/en/FAQ (online, 19. 3. 2008)

[17] VÁCLAVEK Petr. *JavasCript Hotová řešení*. Brno: Computer Press 2006. ISBN: 80-722-6854-6

[18] The IRC Prelude, http://www.irchelp.org/irchelp/new2irc.html (online, 17. 3. 2008)

[19] COLLINS-SUSSMAN Ben, FITZPATRICK Brian, PILATO Michael. *Version Control with Subversion.* USA: O'Reilly 2004. ISBN 10: 0-596-00448-6

[20] The Trac User and Admin. Guide, http://trac.edgewall.org/wiki/TracGuide (online, 20. 3.2008)

[21] phpDoc. Guide, http://manual.phpdoc.org/HTMLframesConverter/default (online, 20. 3. 2008)

[22] Understanding MVC in PHP, http://www.onlamp.com/pub/a/php/2005/09/15/mvc_intro.html (online, 13. 3. 2008)

[23] Viddler API, http://wiki.developers.viddler.com/index.php/Viddler_API (online, 29. 3. 2008)

[24] Using cURL to automate HTTP jobs, http://curl.haxx.se/docs/httpscripting.html (online, 30.3.2008)

[25] Pingback 1.0, http://www.hixie.ch/specs/pingback/pingback (online, 4. 4. 2008)

[26] XML-RPC Specification, http://www.xmlrpc.com/spec (online, 12. 4. 2008)