Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačních technologií

Diplomová práce

Optimalizace databáze pro CMS Drupal

Václav PEKÁREK

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Katedra informačních technologií

Provozně ekonomická fakulta

ZADÁNÍ DIPLOMOVÉ PRÁCE

Pekárek Václav

Informatika

Název práce

Optimalizace databáze pro CMS Drupal

Anglický název

Optimization of database for CMS Drupal

Cíle práce

Diplomová práce je tematicky zaměřena na problematiku využití databáze CMS Drupal. Hlavním cílem práce je optimalizace databáze MySQL pro práci web content management systému Drupal. Dílčí cíle práce jsou:

- analýza CMS Drupal a jeho přístupu k databázi MySQL a
- vypracování přehledu vývoje CMS Drupal.

Metodika

Metodika řešené problematiky diplomové práce je založena na studiu a analýze odborných informačních zdrojů. Praktická část práce je zaměřena na analýzu CMS Drupal a návrhu optimalizace jeho práce s databází MySQL. Pro analytickou část a ověření navrhované optimalizace bude využito zátěžových testů CMS Drupal. Na základě syntézy teoretických poznatků a výsledků praktické části budou formulovány závěry diplomové práce.

Harmonogram zpracování

1. Studium odborných informačních zdrojů, stanovení dílčích cílů a postupu řešení diplomové práce: 06/2013
2. Zpracování teoretických východisek práce: 07/2013 – 09/2013
3. Vypracování vlastního řešení, diskuze a zhodnocení výsledků: 10/2013 – 02/2014
4. Tvorba finálního dokumentu diplomové práce: 03/2014
5. Odevzdání diplomové práce a tezí: 03/2014

Rozsah textové části

60 - 80 stran textu.

Klíčová slova

CMS, redakční systém, Drupal, optimalizace, databáze, MySQL

Doporučené zdroje informací

DUBOIS, P. MySQL Developer's Library. Boston: Addison-Wesley, 2009. ISBN 978-0-678-32938-8.

VANDYK, J. Pro Drupal Development. Apress, 2008. ISBN 978-1430209898.

SCHNEIDER, R. MySQL Oficiální průvodce tvorbou, správou a laděním databází. Praha: Grada Publishing, a.s., 2006. ISBN 80-247-1516-3.

KOFLER, M. Mistrovství v MySQL 5. Brno: Computer Press, a.s., 2007. ISBN 978-80-251-1502-2.

SCHWARTZ, B. MySQL profesionálně: optimalizace pro vysoký výkon. Brno: Zoner Press, 2009. ISBN 978-80-7413-035-9.

Vedoucí práce

Šimek Pavel, Ing., Ph.D.

Termín odevzdání

březen 2015

Elektronicky schváleno dne 29.1.2014

Ing. Jiří Vaněk, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 4.3.2014

Ing. Martin Pelikán, Ph.D.

Děkan fakulty

Čestné prohlášení

Prohlašuji, že svou diplomovou práci "Optimalizace databáze pro CMS Drupal" jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu literatury na konci práce. Jako autor uvedené diplomové práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 30. 3. 2015 _____

Poděkování

Rád bych touto cestou poděkoval vedoucímu práce panu Ing. Pavlu Šimkovi, Ph.D. za rady, připomínky a odborné vedení při zpracování této práce.

Optimalizace databáze pro CMS Drupal

Optimization of database for CMS Drupal

Souhrn

Tato diplomová práce se zabývá optimalizací databáze MySQL pro systém správy obsahu Drupal, popisuje vývoj tohoto systému a analýzu přístupu CMS Drupal k databázi. Tato práce také zahrnuje zhodnocení postupů vedoucí k optimálnímu využití databáze a srovnání jejich efektivity v simulovaném zatížení.

Summary

This master's thesis concerns optimization of database MySQL for content management system Drupal, describes evolution of this system and analysis of CMS Drupal to access the database. This thesis also includes evaluating the processes leading to the optimal use of the database and compared their effectiveness in a simulated load.

Klíčová slova: CMS, redakční systém, Drupal, optimalizace, databáze, MySQL.

Keywords: CMS, editorial systém, Drupal, optimization, database, MySQL.

Obsah

1. Úvod ... 9
2. Cíl práce a metodika ... 10
   2.1. Cíl práce ... 10
   2.2. Metodika ... 10
3. Přehled řešené problematiky ... 11
   3.1. Redakční systém Drupal ... 11
      3.1.1. Projekty využívající systém Drupal ... 12
      3.1.2. Architektura CMS Drupal ... 14
      3.1.3. Jádro systému Drupal a základní moduly ... 15
      3.1.4. Zpracování URL požadavků systémem Drupal ... 19
      3.1.5. Souborová struktura systému Drupal ... 20
      3.1.6. Zhodnocení reálné instalace CMS Drupal 6 ... 21
      3.1.7. Verze systému Drupal ... 22
      3.1.8. Porovnání Drupal 6 vs. Drupal 7 ... 23
   3.2. Základní optimalizace výkonu – cache Drupalu ... 26
      3.2.1. Cache Drupalu - databáze MySQL ... 27
   3.3. Rozšířené možnosti optimalizace výkonu ... 28
   3.4. Třívrstvá architektura ... 31
   3.5. Optimalizace – databázová vrstva ... 32
      3.5.1. DB Maintenance ... 33
      3.5.2. MySQL úložné enginy ... 33
   3.6. Optimalizace – aplikační vrstva ... 33
   3.7. Optimalizace – prezentační vrstva ... 34
   3.8. Pressflow ... 34
   3.9. Jazyk SQL ... 36
   3.10. Databáze MySQL ... 37
      3.10.1. Úvod - historie ... 37
      3.10.2. Architektura databázového serveru ... 37
      3.10.3. Základní typy uložišť ... 38
4. Praktická část ... 41
   4.1. Optimalizace databáze MySQL pro práci CMS Drupal ... 41
      4.1.1. Volba a vliv typu uložiště databáze MySQL ... 41

4.1.2. Optimalizace uložiště MyISAM ... 42
4.1.3. Replikace databáze MySQL ... 44
4.1.4. MySQL Cluster ... 49
4.1.5. Optimalizace dotazů ... 51
4.2. Implementace Pressflow 6 ... 53
4.2.1. Replikace databáze s Pressflow 6 ... 53
4.2.2. Profilování Drupal 6 a Pressflow 6 ... 53
4.2.3. Bezpečnost Pressflow ... 55
4.3. Způsoby testování ... 55
4.3.1. Rozdělení zátěžových testů ... 56
4.4. Použité způsoby testování ... 57
4.4.1. AuthcacheFooter – statistika průběhu generování stránky ... 57
4.4.2. Apache Benchmark ... 57
4.4.3. Apache JMeter ... 57
4.5. Ověření navrhované optimalizace – zátěžové testy ... 58
4.5.1. Test zátěže serveru - Apache Benchmark ... 58
4.5.2. Test zátěže serveru – Apache JMeter ... 58
5. Zhodnocení výsledků a doporučení ... 62
6. Závěr ... 65
7. Seznam použitých zdrojů ... 66
8. Přílohy ... 69
8.1. Seznam obrázků a grafů ... 69
8.2. Seznam tabulek ... 70
8.3. Seznam příloh ... 70

1. Úvod

Redakční systémy umožňují kompletní správu obsahu webu. Umožňují uživatelsky přijatelnou cestou bez znalosti programování a značkovacích jazyků publikovat nejen texty, ale i jiné typy obsahu. Umožňují instalovat doplňující části, doplňky již realizovaných řešení, jako jsou fotogalerie, pokročilá diskuzní fóra, nebo celé elektronické obchody.

Pro vývojáře představuje použití redakčních systémů rozdělení vývoje do více částí za použití šablon a doplňků. Komunita uživatelů a vývojářů určitého redakčního systému se podílí na společném vývoji mnoha doplňků, jejichž znovupoužitelnost zefektivňuje vývoj celé řady systémů.

Systém Drupal patří mezi nejznámější trojici redakčních systémů WordPress a Joomla! Architektura CMS Drupal je z výše jmenovaných nejrobustnější, jeho využití je jak v malých projektech, tak i v rozsáhlých systémech s mnoha uživateli. Umožňuje nejen vytvořit pro méně zkušeného uživatele relativně snadno osobní stránky nebo blog, ale i velmi složitou webovou aplikaci pro státní správu, např. portál městského úřadu (Město Limerick, Irsko), univerzitní web (Dublin City University), nebo stránky Bílého domu.

CMS Drupal se svým charakterem open-source, osvědčil v oblasti podporující technologie, otevřená data, znalosti a otevřený přístup v rezortu zemědělství. Specifická distribuce AgriDrupalu představuje nástroj, který je např. vhodný pro využití v rozvojových zemích. Zcela odlišné použití sytému Drupal představuje portál Zemědělských univerzitních novin (iZUN.eu).

Diplomová práce je zaměřena na problematiku využití databáze systémem Drupal. V případech použití s vysokým vytížením systému je právě databáze pro redakční systém hlavním uložištěm dat. Efektivní chod databáze představuje jeden z klíčových aspektů pro celkový výkon redakčního systému. CMS Drupal je velmi často provozován s open-sourceovou databází MySQL, na jejíž optimalizaci se zaměřuje tato práce.

2. Cíl práce a metodika

2.1. Cíl práce

Diplomová práce je tematicky zaměřena na problematiku využití databáze CMS Drupal. Hlavním cílem práce je optimalizace databáze MySQL pro práci web content management systému Drupal.

Dílčí cíle práce jsou:

- analýza CMS Drupal a jeho přístupu k databázi MySQL
- vypracování přehledu vývoje CMS Drupal

2.2. Metodika

Práce čerpá ze zdrojů odborné literatury a informačních zdrojů zveřejněných na internetu, uvedených v seznamu použité literatury. Praktická část práce je zaměřena na analýzu CMS Drupal a zhodnocení možností optimalizace práce s databází MySQL. Pro analytickou část, ověření navrhovaných optimalizačních nástrojů a jejich porovnání, bude využito zátěžových testů CMS Drupal. Na základě syntézy teoretických poznatků a výsledků praktické části budou formulovány závěry diplomové práce.

V rámci studia odborných informačních zdrojů v teoretické části bude navázáno na zpracované téma bakalářské práce *Zabezpečení a správa přístupů k databázi MySQL*.

Komplexní analýza CMS Drupal bude zahrnovat analýzu možností optimalizace zdrojového kódu systému souvisejícího s přístupem k databázi. Analýza bude také v některých částech obecně souviset s výkonem systému jako celku. Dále bude zahrnovat zhodnocení možností zvýšení výkonu, respektive odezvy v souvislosti s počtem uživatelů modelového webového serveru.

V práci se vychází z instalace CMS Drupal 6 a databáze MySQL verze 5.6 na platformě LINUX/Ubuntu, není-li uvedeno jinak.

3. Přehled řešené problematiky

Tato kapitola popisuje redakční systém CMS Drupal. Obsahuje základní popsání charakteristik redakčního systému a jednotlivých verzí. Je rozebrán vývojový cyklus a odlišnosti mezi sledovanými verzemi. Na konkrétních příkladech je uvedeno použití systému Drupal. Zahrnuje popis architektury CMS Drupal a celé aplikace, hodnotí způsoby optimalizace. Dále také obsahuje charakteristiku databáze MySQL a její architekturu.

3.1. Redakční systém Drupal

Projekt redakčního systému začal vznikat v roce 1998. Autorem je holandský student Dries Buytaert. Projekt nejdříve sloužil ke sdílení informací v rámci školy. Po ukončení B. studia, od roku 2000, začal Buytaertův systém sloužit pro komunikaci s bývalými spolužáky a kolegy v podobě fóra. Původně se systém jmenoval Drop[1], první veřejná verze byla už pojmenována Drupal[2]. Společně s uvedením první verze byly zveřejněny i zdrojové kódy – projekt je šířen pod licencí GNU General Public Licence[3] (Drupal.org, 2014).

Na vývoji systému se kromě Driese Buytaerta nyní podílejí stovky programátorů jádra. Dále i správa jednotlivých verzí je přidělena daným vedoucím. Na doplňkových modulech, dostupných z portálu Drupal.org, pracují ostatní programátoři celé komunity.

Drupal je velmi přizpůsobivý, obecně se jedná o Framework pro zprávu obsahu (Content Management Framework – CMF), z něhož je možné sestavit systém pro zprávu obsahu (Content Management System – CMS) přesně na míru podle potřeb. (Drupal not CMS, 2014)

Prvním významným úspěchem systému Drupal, které napomohlo rozšíření systému, bylo v roce 2003 použití Drupalu k vytvoření „DeanSpace“, stránek Howarda Deana, kandidáta na amerického prezidenta v primární volbě.

[1] z holandštiny - vesnice
[2] z holandštiny - drop (dle anglické výslovnosti *druppel*) (2014)
[3] GNU GPL – licence pro svobodný software

3.1.1. Projekty využívající systém Drupal

S použitím CMS Drupal se lze setkat na celé řadě webových stránek, včetně prestižních stránek Bílého domu (*whitehouse.gov*), Universal Music (*universalmusic.com*), nebo stránkách měst např. Limerick, Irsko (*limerick.ie*). V České republice je využíván CMS Drupal např. televizí Prima (*iprima.cz*) a od roku 2014 i Českým rozhlasem (*rozhlas.cz*).

Zemědělské univerzitní noviny - iZUN

Webové stránky univerzitních novin České zemědělské univerzity iZUN (Zemědělské univerzitní noviny) jsou provozovány na CMS Drupal 6. Redakční systém pro iZUN sdružuje obě funkce, tedy slouží jak pro editování a přípravu vydávaného obsahu, tak i pro publikaci obsahu.

V oblasti zpravodajství se lze setkat s tzv. modelem *Otevřené žurnalistiky*[4]. Tento přístup klade zvýšené nároky na redakční systém. Umožňuje čtenáři aktivně se podílet na přípravě obsahu. V případě iZUN.eu se jedná o uživatele anonymní i přihlášené. Anonymní uživatelé mohou přidávat obsah následujících typů:

Aktuality – umožňují kterémukoli (i nepřihlášenému) uživateli vkládat přímo z hlavní stránky aktuality obsahující datum, nadpis (název), obsah, fotografie a kontaktní e-mail.

Fotografie, obrázky – možnost aktuálně vkládat fotografie, například z mobilních zařízení, pro kteréhokoli uživatele.

Podněty a reakce k připravovaným článkům – příprava materiálů k tvorbě článků, které jsou zasílány čtenáři do redakčního systému.

Registrovaný – přihlášený uživatel může využít pokročilé funkce Otevřené žurnalistiky a například založit *Blog*. V redakčním systému se jedná o dalšího redaktora, ale s omezenými právy určenými pouze pro specifické vlastnosti *blogu*. Těmito

[4] Otevřená žurnalistika je trend, který využívá možnosti informačních zdrojů (např. sociální média), umožňuje aktivní zapojení čtenáře a jeho spolupráci. Tuto možnost, zapojit čtenáře do aktivního procesu vzniku článku, využily mezi prvními švédské noviny Norran a později i britský list The Guardian. (http://www.izun.eu/otevrenazurnalistika, 15. 12. 2014)

vlastnostmi jsou omezené možnosti vkládání obsahu pouze na prostý text, možnost vložení videoobsahu ze zdroje youtube.com a omezený počet fotografií v galerii.

O setrvání na verzi Drupalu 6 bylo rozhodnuto z důvodu náročnosti migrace mezi jednotlivými verzemi Drupalu a značné velikosti celého webu obsahujícího celou řadu vlastních modulů kompatibilních pouze s verzí 6.x.x. Přechod na novou aktuální verzi 7 by byl značně náročný a nezaručoval by pozdější kompatibilitu s očekávanou verzí Drupalu 8. Ta by se měla výrazně odlišovat od předešlých verzí. Z tohoto důvodu je ponechána verze CMS Drupal 6 a plánován pozdější přechod na zcela odlišnou verzi 8 (na problematiku kompatibility a verzování je zaměřena samostatná *kapitola 3.1.7 Verze systému Drupal*).

AgriDrupal

AgriDrupal je speciální sestava CMS Drupal, která je spravována a dostupná na portálu AIMS [5] (Agricultural Information Management Standards). Portál AIMS je součástí celosvětového hnutí pro otevřené zemědělské znalosti CIARD (Coherence in Information for Agricultural Research for Development). AgriDrupal vytváří standardní nástroj pro sdílení a správu znalostí v oblasti zemědělství. Hlavním přínosem v informačním managementu je stanovení standardů, jejichž přijetí a údržba je dlouhodobě udržitelná. Z těchto důvodů vznikl AgriDrupal, který využívá silné komunitní podpory tradičního open source CMS Drupal.

AgriDrupal platforma umožňuje podporu vzniku webových projektů organizací spojených se zemědělstvím po celém světě. (AgriDrupal, 2014)

- Agropedia Indica – národní portál zemědělství v Indii, znalostní model.
- ILRI [6] - organizace zabývající se potravinovou soběstačností a redukcí chudoby rozvojových zemí.

[5] Agricultural Information Management Standards je portál podporující standardy, technologie, otevřená data, znalosti a otevřený přístup v oblasti zemědělství. Působí v těchto oblastech: Information Management Resources, doporučení v Information Management, komunitní praxe a rozvoj odborníků. AIMS je součástí širší organizace CIARD, celosvětového hnutí pro otevřené zemědělské znalosti. (AIMS, 2014)

[6] International Livestock Research Institute (https://www.ilri.org/, 20. 12. 2014)

- IFPRI[7] - organizace hledající trvalá a udržitelná řešení pro ukončení hladu a chudoby.
- ICRAF[8] - organizace, jejímž cílem je snaha o zmírnění odlesňování a ničení půdy v tropických oblastech.
- e-agriculture – celosvětová komunita, jejímž cílem je výměna informací a zdrojů pro využití ICT (komunikačních technologií) pro udržitelné zemědělství a rozvoj venkova.
- World Food Programme – světový potravinový program, který se zabývá potravinovou bezpečností.
- CIARD - celosvětové hnutí pro otevřené zemědělské znalosti.
- KAINeT[9] – iniciativa pro otevřený přístup k informacím zemědělského výzkumu v Keni.

Sestava AgriDrupal neobsahuje primárně části s vlastním softwarovým vývojem, ale převážně je tvořena z běžných vhodných modulů a nastavení. Vlastní programování a vývoj je co nejvíce minimalizován.

3.1.2. Architektura CMS Drupal

Popis architektury CMS Drupal je zaměřen na verzi Drupalu 6, na kterou je zaměřena tato práce.

CMS Drupal 6 se skládá z řady PHP skriptů (jádra systému a modulů), souborů JavaScriptu a kaskádových stylů, konfiguračních souborů a dalších. Pro ukládání dat je podporována databáze MySQL a PostgreSQL.

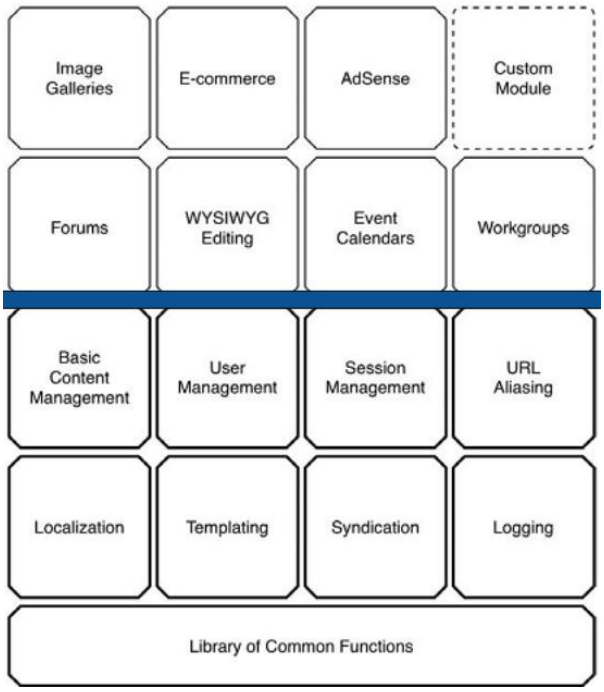
V základní konfiguraci lze spravovat účty uživatelů, nastavení oprávnění k přístupu a přidělení uživatelských rolí. Lze přidávat a upravovat obsah a spravovat *témata* zobrazení. Dále je k dispozici správa *bloků,* jejichž prostřednictvím lze upravovat obsah v jednotlivých *regiónech* stránek. Menu je k dispozici ve více typech (*primární, sekundární*). V základní konfiguraci je také možnost kategorizace obsahu (modul

[7] International Food Policy Research Institute (http://www.ifpri.org/, 20. 12. 2014)
[8] International Center for Research in Agroforestry (http://www.ciesin.org/IC/icraf/icrafmore.html, 20. 12. 2014)
[9] Kenya Agricultural Information Network

taxonomy) s využitím obvyklých způsobů pomocí kategorií, tagů nebo uživatelem definovaných slovníků.

3.1.3. Jádro systému Drupal a základní moduly

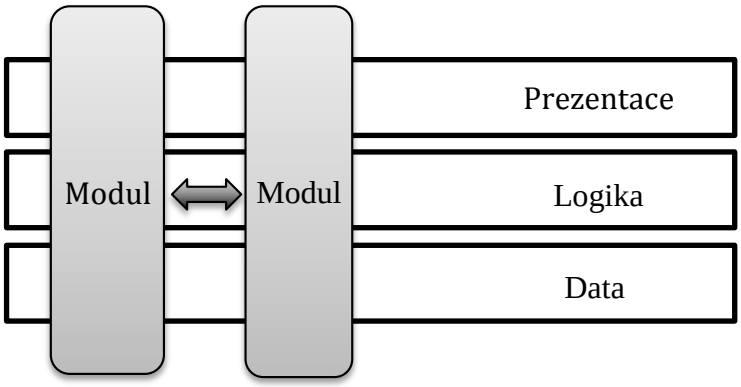
Na *obrázku č. 1: Základní přehled architektury CMS Drupal* jsou graficky znázorněny základní části systému CMS Drupal 6. Horní polovina uvedené grafiky obsahuje volně dostupné moduly rozšíření (např. zde uvedené moduly pro Forum, WYSIWYG Editor, AdSense aj.) a také vlastní moduly. Spodní část obsahuje moduly jádra systému, základní subsystémy a knihovny. Jakékoliv zásahy do této části jsou přísně zakázány a mohly by ohrozit stabilitu a bezpečnost celého systému.



Obrázek č. 1: Základní přehled architektury CMS Drupal (VanDyk, 2008)

Architektura CMS Drupal 6 je vzoru PAC (Presentation Abstraction Control). Každý modul obsahuje vlastní celou strukturu všech tří částí. Z tohoto důvodu jsou vlastně moduly uspořádány vertikálně a nejsou uspořádány horizontálně - *obrázek č. 2: Architektura CMS Drupal (Explaining architectural tiers Drupal, 2014)*. Každý modul může obsahovat API rozhraní, které umožňuje přístup dalších modulů k funkcím daného modulu, a tím i začlenění těchto funkcí do ostatních modulů. Tato unikátní architektura

v Drupalu nazývaných háků [10] umožňuje komunikaci jádra systému s moduly a také komunikaci modulů mezi sebou.



*Obrázek č. 2: Architektura CMS Drupal (*Explaining architectural tiers Drupal, 2014*)*

Systém CMS Drupal 6 a starší verze byly výhradně procedurální. Jedinou výjimku tvoří ve verzi 6 *xml parser*. Verze CMS Drupal 7 zachovává procedurální jádro systému, včetně *háků*, obsahuje však i objektové subsystémy. Připravovaná verze CMS Drupal 8 obsahuje již zcela objektovou architekturu. (VanDyk, 2008)

Drupal je tedy modulární systém, lze přidávat a spravovat celou řadu nejrůznějších modulů. Prvotní instalace obsahuje několik výchozích, resp. povolených modulů, tvořících malé stabilní a výkonné jádro systému. Řada dalších modulů je připravena k zapnutí, např. *komentáře, vyhledávání, fóra, ankety aj*.

Další moduly, včetně všech verzí, lze vyhledat a stáhnout z webových stránek Drupal.org (*http://drupal.org/project/modules*). Zde je k dispozici více jak 7 000 modulů pro Drupal 6. (Drupal.org, 2014) Na uvedeném serveru je také soustředěn vývoj těchto modulů, aktualizace a podpora ve formě diskuzních fór, popis a dokumentace využití modulů. Vzhledem k velkému počtu členů celé komunity[11] (1 152 000 lidí v 229 zemí, 180 jazyků), jak vývojářů (37 000), tak i uživatelů, je tento model komunitního přístupu velmi úspěšný (Drupal.org, 2014).

Informace o zabezpečení systému Drupal a jednotlivé záznamy o stabilitě systému, systémové události a chyby se ukládají do databáze. CMS Drupal v pravidelných

[10] V originále angl. *Hooks*.
[11] Celá komunita pro CMS Drupal bez rozlišení verzí.

intervalech zjišťuje aktuálnost jádra a instalovaných modulů sám, v případě nové aktualizace, nebo dostupné kritické bezpečnostní aktualizace, doporučuje administrátorovi provést nezbytnou aktualizaci. Dopodrobna se tomuto tématu věnuje *kapitola 3.1.8 Aktualizace*.

Adresa stránek URL, čistá URL

Systém Drupal umožňuje pracovat s čistými[12] adresami URL, které jsou vhodné pro vyhledávače a hlavně uživatelsky přívětivé. V současnosti je tento formát URL adres nepsaným standardem. Lze také nastavit více různých adres k dané stránce.

Šablony

Silnou stránkou systému Drupal je podpora celé řady profesionálně připravených témat/šablon, které jsou dostupné zdarma (více jak 2000 šablon (Drupal.org, 2014)), nebo jsou placené a zahrnují i technickou podporu[13].

Šablona je kolekce souborů zajišťujících prezentační vrstvu systému. Systém Drupal, tedy jeho jádro, využívá šablony skrze *theme engine*. Základní instalace obsahuje několik předpřipravených výchozích šablon.

Šablony lze upravovat a konfigurovat v administraci systému Drupal. Lze upravovat vzhled webu (barvu a velikost písma - pokud to šablona podporuje, lze změnit i barevné schéma), rozmístění prvků webu (menu, bloky aj.).

Bloky

Bloky tvoří velmi důležitý prvek webu, postaveným na systému Drupal. Rozmístění bloků na stránce webu určuje funkční a informační charakter celého webu. Obsah, kterým může být seznam článků, anketa, menu, reklama a jiné, je umístěn v různých blocích v několika typických částech stránky. Zpravidla se jedná o tyto regiony: hlava, levý nebo pravý sloupec, obsah a pata stránky. Tyto části stránky jsou definovány v šabloně a v šabloně může být určeno i mnohem více regionů.

[12] Čistá URL adresa obsahuje pouze alfanumerické znaky a lomítka. Nevyskytuje se tvar: /?=q.
[13] Příkladem komerčních šablon s 24/7 podporou http://www.templatemonster.com/drupal-themes.php, 5. 10. 2014.

Typy obsahu

Drupal obsahuje vestavěnou podporu pro volbu uživatelem definovaných typů obsahu – přizpůsobení bez programování. Každý typ obsahu je popsán vhodnými metadaty.

Standard slovníků, propojení CCK fields s existujícími RDF vlastnostmi a určení RDF třídy pro daný typ obsahu.

Uzel - node

Každý obsah je vložen do tzv. uzlu, který tvoří jednotku obsahu. To znamená, že když vzniká nový uzel, je zvolen typ obsahu podle předem definovaných typů uložených v systému. Uzel je identifikován specifickým číslem *nid*, se kterým lze dále jednotně pracovat skrze jiné moduly.

Kategorie obsahu

Modul Taxonomy umožňuje vytvářet kategorie. Terminologie použitá v Drupalu je specifická a skládá se z následujících pojmů. Nejvýše nadřazen je tzv. *slovník* kategorií, jednotlivé kategorie jsou jemu podřazené. Pro jednotlivé názvy kategorií ve slovníku je použito označení *termín*.

Kategorie obsahu lze vhodně využít při vytváření menu, případně pro navigaci na webové stránce. Drupal obsahuje tři druhy menu. Prvním je *navigace,* využívaná při administraci a dalšími jsou *primární* a *sekundární menu*. Primární menu se zpravidla využívá pro hlavní navigaci uživatele na webové stránce. Sekundární menu s ostatními, méně důležitými odkazy, bývá zobrazeno například v patičce webové stránky.

Pohledy (Views)

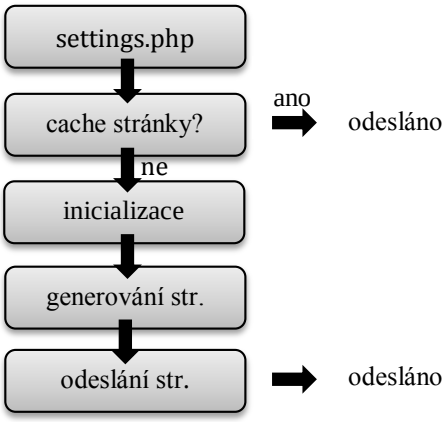
Drupal umožňuje přes tzv. pohledy zpracovat jakýkoli obsah. Obsah je filtrován podle celé řady kritérií, např. typ obsahu, kategorizace aj. Tento modul také umožňuje vytvořit sestavy pro export. Vybraný obsah je možné vkládat do bloků a zobrazovat na vybraných částech stránky (viz Bloky).

Uživatelé

Systém Drupal umožňuje podrobně spravovat systém přístupových práv podle uživatelských rolí (anonymní uživatel, přihlášený uživatel, redaktor a další volně editovatelné role). S jednotlivými uživateli, respektive s jejich nastavením, lze dále pracovat: zobrazit a upravit údaje, odstranit a přidat uživatele. K již existujícím uživatelům lze přiřadit role obsahující příslušné oprávnění. Role, respektive jednotlivá oprávnění, je možné definovat a upravovat. V neposlední řadě je také možné přesně určit možnosti postupu, jak se mohou uživatelé registrovat (např. proces schvalování nového uživatele).

3.1.4. Zpracování URL požadavků systémem Drupal

Drupal obecně zpracovává HTML požadavky podle následujícího schématu.



Obrázek č. 3: Přehled zpracování DRUPAL HTTP požadavku (Hodgdon, 2012)

Nejprve je určen soubor s nastavením, který se načte a provede. V dalším kroku systém Drupal rozhoduje, zda použije cache stránky podle typu uživatele. Pokud je uživatel anonymní, zkontroluje se cache stránek, zda je stránka v této cache uložena, pak je poskytnuta a proces ukončen. Cache stránek není využita, jedná-li se o přihlášeného uživatele. Podrobněji se problematikou paměti cache zabývají následující kapitoly této práce (*kapitola 3.2.1 Cache Drupalu - databáze MySQL*).

Není-li možné použít paměť cache, následuje inicializace a je provedeno nastavení systémových proměnných, jazykových systémů a modulů jádra systému Drupal, případně ostatních povolených modulů. Pokud je systém v režimu offline, například je prováděna změna nebo údržba, je zobrazena informační stránka *údržby*. Je-li systém online a přístup

uživatele je autorizován, jsou provedeny požadované funkce a generován obsah pro daný požadavek. Výsledkem je předrenderovaný obsah, který je následně podle zvolené metody doručení obsahu dále zpracován. Je vytvořena http hlavička, obsah je vyrenderován podle příslušného tématu do jazyka HTML a odeslán webovému serveru.

3.1.5. Souborová struktura systému Drupal

Stažený a rozbalený CMS Drupal 6.x.x obsahuje složky a soubory uvedené na *obrázku č. 4: Souborová struktura CMS Drupal*. Pro instalaci databáze jsou důležité zejména soubory *INSTALL.mysql* pro databázi MySQL a *.pgsql* pro databázi PostgreSQL. Instalace databáze může být provedena automaticky skriptem *install.php,* který provede celou instalaci systému Drupal. Instalační průvodce nastaví přístupové parametry k databázi, heslo, název webového projektu, e-mailovou adresu, časové pásmo aj. Při této instalaci dojde ke zkopírování[14] souboru *sites/default/default.settings.php*, který obsahuje základní parametry nastavení přístupu k databázi. Nastavení bezpečnosti, tedy přístupová práva pro jednotlivé složky, stejně jako podrobný postup instalace, nejsou předmětem této práce[15].

Všechny soubory a složky kromě *sites/* uvedené na *obrázku č. 4: Souborová struktura CMS Drupal* jsou součástí jádra Drupalu a nesmí se měnit. (Drupal.org, 2014)Jedinou výjimku představují soubory *.htaccess*, *robots* a složka *profiles/*. Vlastní moduly a šablony se ukládají do *sites/modules,* respektive *sites/themes*. Složka *sites/* obsahuje i jiná uživatelská data, např. soubory ke stažení a obrázky.

[14] V některých případech, v závislosti na nastavení hostingu, je nutné tento krok provést ručně.
[15] Zabezpečení je podrobněji uvedeno: https://www.drupal.org/server-permissions.

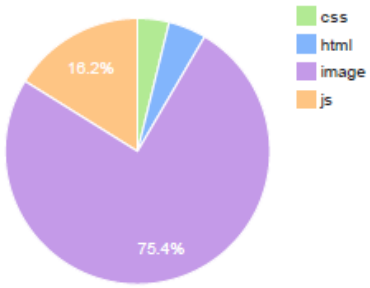
```
Drupal-6.xx/
├──── includes/
├──── misc/
├──── modules/
├──── profiles/
├──── scripts/
├──── sites/                          #moduly, šablony uživ.
│     └────── all/
│     └────── default/
│        └──── default.settings.php  #obsahuje nastavení DB
├──── themes/
├──── .htaccess
├──── COPYRIGHT
├──── cron.php
├──── CHANGELOG
├──── index.php
├──── INSTALL
├──── INSTALL.mysql
├──── INSTALL.pgsql
├──── install.php                   #průvodce instalace CMS
├──── LICENSE
├──── MAINTAINERS
├──── robots
├──── update.php                    #update CMS na novou verzi
├──── UPGRADE
├──── xmlrpc.php
```

Obrázek č. 4: Souborová struktura CMS Drupal

3.1.6. Zhodnocení reálné instalace CMS Drupal 6

Zhodnocení stávající instalace CMS Drupal 6 pro stránky iZUN.eu bylo provedeno nástrojem *Web Page Performance Test*[16]. Tento nástroj umožňuje analyzovat proces načtení webové stránky, výstupem je tzv. vodopádový model, tj. časová posloupnost jednotlivých http požadavků a stažených souborů. Zaznamenává dobu pro DNS vyhledání, navázání spojení, TTFB (Time to First Byte) a samotné stažení. Podrobný grafický výstup je obsažen v *příloze č. 1: Načtení stránky izun.eu, vodopádový model načítání (měření 10. 1. 2015 http://www.webpagetest.org/)*, který popisuje jednotlivé http požadavky všech připojení. V daném případě měření byl celkový čas potřebný k načtení stránky 4,7s, obsahoval 131 požadavků a celková velikost byla 883 KB. Celá řada http požadavků je ale spouštěna souběžně více spojeními. Zobrazení všech 26 spojení je v *příloze č. 2: Připojení, vodopádový model připojení (měření 10. 1. 2015 http://www.webpagetest.org/).*

[16] http://www.webpagetest.org/ (10. 1. 2015)



MIME Type	Requests
image	98
js	21
html	6
css	5
flash	0
font	0
other	0

Obrázek č. 5: Požadavky podle typu MIME (měření 10. 1. 2015 http://www.webpagetest.org/)

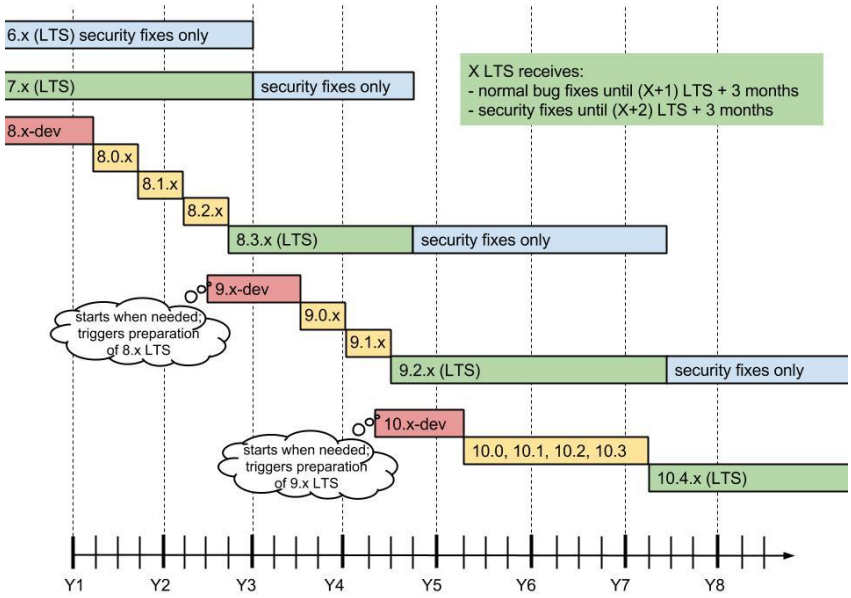
Na uvedeném o*brázku č. 5: Požadavky podle typu MIME (měření 10. 1. 2015 http://www.webpagetest.org/)* je procentuální podíl všech načtených typů souborů podle MIME. Největší podíl 75,4% činily grafické soubory typu *image,* s druhým největším podílem 16,2% byly soubory Javascriptu (typ *js*). Soubory typu *css* kaskádových stylů a *html,* které zaujímaly pouze 3,8%, respektive 4,6% pro *html* soubory. Test byl proveden pouze pro načtení úvodní stránky a v době špičky vytížení serveru.

3.1.7. Verze systému Drupal

Systém Drupal používá pro označení jednotlivých verzí tzv. Sémantické verzování 2.0.0, které se skládá ze tří čísel oddělených tečkou. První číslo (MAJOR verze) rozlišuje verzi, která není zpětně kompatibilní s nižšími verzemi, tedy obsahuje takové změny, které tuto kompatibilitu vylučují. Druhé číslo (MINOR verze) odlišuje přidání nové funkcionality, která je zpětně kompatibilní. Poslední číslo (PATCH verze) značí opravení chyb a zachovává také zpětnou kompatibilitu (Preston-Werner, 2015).

Od roku 2013 je dostupný plán vývojového cyklu systému Drupal (*obrázek č. 6: Vývojový cyklus jádra CMS Drupal*), který začíná verzí 8.x-dev a pokračuje dále i pro verze 9 a 10. Nový návrh vývojového cyklu zahrnuje i ukončení podpory Drupalu 6.x, v režimu LTS[17] a i stávající verze 7.x v LTS.

[17] LTS - Long Term Support, dlouhodobá podpora.



Obrázek č. 6: Vývojový cyklus jádra CMS Drupal (Buytaert, 2015)

Nový přístup k vývojovému cyklu změnil způsob uvádění nového Drupal 8. Každých šest měsíců je uváděna nová verze 8.1, 8.2, 8.3 atd., která může obsahovat nové vylepšení. Pouze poslední uvedená verze je v režimu dlouhodobé podpory. Po ukončení dlouhodobé podpory jsou zveřejňovány pouze bezpečnostní opravy.

Aktuální verze Drupal 7.x podporuje v základní konfiguraci databázi MySQL 5.0.15 a vyšší (případně obdobnou databázi MariaDB od verze 5.1.44), PostgreSQL 8.3 a SQLite 3.x (2014). Nejnovější verze 8.x (beta) není ještě plně stabilní, z tohoto důvodu se tato práce zabývá následujícím srovnáním verzí 6.x a 7.x.

3.1.8. Porovnání Drupal 6 vs. Drupal 7

Na první pohled viditelné změny se týkají ovládacích prvků administrace. Přibyla lišta [18] s administračním menu v horní části stránky. Rozšířena byla také zpráva jednotlivých bloků na stránce, nyní je možné je upravit přímo přes odkaz v příslušném bloku. Při správě nových modulů a témat, respektive přidávání modulů a témat, není nutné je nahrávat přes FTP klienta, ale je možné je přidat přímo z administrace systému. Do administračního rozhraní byla také přímo integrována podpora instalace vizuálních

[18] Podobné modulu Administration menu (http://drupal.org/project/admin_menu). (Polzer, 2011)

editorů textu (WYSIWYG[19]). V předešlých verzích systému Drupal byla instalace těchto editorů značně obtížná a vyžadovala zkušenosti s implementací WYSIWYG editorů.

Modul *Content Constuction Kit* (CCK), umožňující přidávat a spravovat vlastní vstupní pole, známý z předešlých verzí, byl integrován jako modul *Fields*. Stejně byl i integrován modul pro podporu Resource Description Framework (RDF), který umožňuje standardizovaný formát popisu zdrojového dokumentu. Tento modul je nezbytnou součástí i AgriDrupalu (viz *kapitola 3.1.1 Projekty využívající systém Drupal*).

Častým nedostatkem předešlých verzí byla absence jednotného nástroje pro práci s obrázky. Zejména nahrávání, následná vhodná úprava obrázků pro webovou prezentaci a jejich uložení na serveru. Tyto činnosti byly vykonávány celou řadou doplňkových modulů (*Upload Image, Image module, Imagefielad aj.*). Drupal 7 ale integruje modul *ImageCache*, který zmenšuje a optimalizuje obrázky přidávané s obsahem. (Polzer, 2011)

Drupal 7 obsahuje testovací prostředí, které je možné příslušným modulem spustit a před ostrým nasazením web otestovat. (Polzer, 2011) Pro komunikaci s databází je vyžadováno rozšíření PDO[20] a z toho vyplývající požadavek na minimální verzi PHP 5.2. Významný rozdíl je i v použití úložných enginů. Místo dřívějšího MyISAM je použitý výchozí engine InnoDB u databáze MySQL. Výhody tohoto úložného enginu jsou shrnuty v *kapitole 3.10.3 Základní typy uložišť.*

Novinkou v této verzi je i podpora práce s příkazovou řádkou, umožňující instalaci Drupalu a následnou správu. Například Drush[21] (Drupal Shell), vytvářející rozhraní pro správu přes příkazovou řádku; velice užitečný nástroj v profesionálním použití při správě rozsáhlejšího webu, který usnadní práci a odstraní nutnost využití v některých případech nepohodlného grafického rozhraní.

Souborový systém Drupalu byl přepracován a rozšířen o jednodušší podporu externích zdrojů souborů. Je možné propojení například se službou Flicker pro správu obrázků, nebo se službou Amazonu jako cloudovým uložištěm pro správu souborů.

[19] Akronym What You See Is What You Get, tedy Co vidíš, to dostaneš.
[20] PDO – PHP Data Objects, objektové rozhraní databází pro PHP.
[21] Drush, http://docs.drush.org/en/master/,10. 12. 2014

Využití Drupalu má své důležité místo také pro firemní web. Jan Polzer v knize Drupal 7 Podrobný průvodce tvorbou a správou webů uvádí čtyři základní požadavky, které systém Drupal splňuje:

- řešení nezávislé na jednom dodavateli
- možnost přenosu webu na libovolný server (míněno nezávislost serveru dodavatele)
- možnost aktualizace obsahu svépomocí, bez dodavatele webu
- zabudovanou onsite optimalizaci pro vyhledávače – SEO

(Polzer, 2011, str. 27)

Přechod mezi verzemi

Přechod mezi jednotlivými verzemi Drupalu je poměrně komplikovaný. Mezi verzemi není zajištěna kompatibilita. Jednotlivé moduly, šablony (témata), jsou vždy sestaveny pro danou MAJOR verzi a případně MINOR pod verzi s ní kompatibilní.

Pro upgrade Drupalu na novou verzi je možné ověřit dostupnost odpovídajících verzí instalovaných modulů v modulu *Upgrade Status*[22]. Tento modul zobrazí dostupnost odpovídající nové verzi modulů pro novou verzi jádra Drupalu.

Příklad zpoždění dostupnosti modulů pro novou verzi uvádí Jan Polzer na příkladu přechodu na verzi Drupalu 6: *„…po uvolnění Drupalu 6 trvalo tvůrcům modulů CCK a Views, tedy modulům často označovaným jako nejpoužívanější, asi půl roku, než byli schopni nabídnout stabilní kompatibilní verzi pro Drupal 6“.* (Polzer, 2010, str. 30)

Aktualizace

Do verze Drupalu 6 bylo vždy nutné nahradit stávající soubory novými aktuálními soubory (jádro systému, moduly, šablony) a poté spustit script *update.php* a aktualizovat databázi. Před samotnou aktualizací je ale vždy nezbytné celý systém zálohovat, včetně databáze.

[22] Upgrade Status modul (http://drupal.org/project/upgrade_status)

Od verze Drupalu 7 se aktualizace provádí v administraci, kde jsou červeně označeny bezpečnostní aktualizace a žlutě označeny zcela nové verze upravující funkce modulů. Tyto opravy a aktualizace lze provést příslušným tlačítkem *Stáhnout aktualizace*. Proces je obdobný jako při instalaci nových modulů. Opět je doporučeno předem provést zálohu databáze a aktualizovaných částí systému. Aktualizace je dokončena jako v předešlých verzích Drupalu spuštěním scriptu *update.php*.

3.2. Základní optimalizace výkonu – cache Drupalu

Administrativní rozhraní Drupalu umožňuje nastavit několik různých úrovní cachování. Tyto úrovně odpovídají výši přínosu, tj. zrychlení získání odpovědi – načtení stránek pro uživatele. Do cache se ukládá obsah, který je zobrazen anonymnímu, tedy nepřihlášenému návštěvníkovi. Obecně je ukládán obsah, který se nemění často (*proměnné, sessions, menu, bloky, nody-stránky, views*). Cache je uložena na serveru v dočasném adresáři a v databázi.

Rozlišujeme tři stavy nastavení cache: *vypnutá, normální a agresivní*. Normální režim je doporučen pro běžné produkční použití. Neměl by ovlivňovat správnou funkčnost webu a případně negativně interagovat s instalovanými moduly. Při použití agresivního nastavení cache je nutné individuální odzkoušení pro různé nastavení stránek a kombinaci povolených a instalovaných modulů[23].

Pokud dojde ke změně obsahu, je nutné obnovit cache. Aby nedocházelo k této obnově příliš často, nebo naopak málo často, je nutné zvolit odpovídající minimální životnost cache. Pokud ale dojde ke změně obsahu během nastavené prodlevy, není tato změna aktualizována v paměti cache. (Trevor, 2010)

Pro další optimalizaci v základní instalaci systému Drupal je dostupný nástroj komprese stránek[24], který omezuje objem přenášených dat mezi webovým serverem. Ve výsledku zrychlí komunikaci s uživatelem.

[23] Kontrola kompatibility s agresivním nastavením cache pro každý modul zvlášť.
[24] Komprese stránek musí být podporována hostingovým serverem.

Optimalizace CSS souborů je vhodné využít, pokud šablona použitého tématu obsahuje více CSS souborů z různých adresářů. Drupal vytvoří dočasný CSS soubor, obsahující všechny kaskádové styly z různých souborů v jediném.

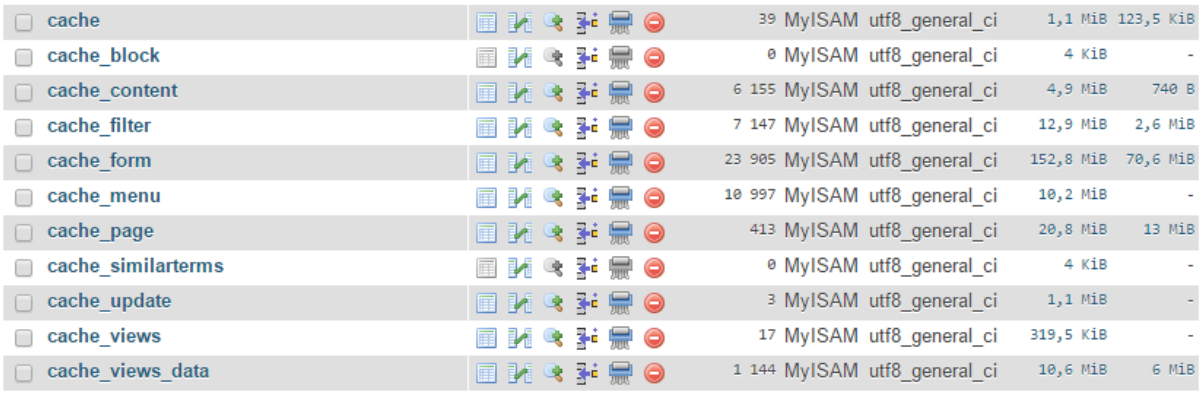
Optimalizace JavaScriptu může způsobit problémy s některými moduly. V produkčním nasazení je nutné zkontrolovat funkčnost s instalovanými moduly a hlavně s moduly v neukončeném vývojovém cyklu (*develop*).

Při konfiguraci, údržbě a ladění webu je nutné všechny výše uvedené optimalizační volby vypnout. V nastavení *výkonu* (*Performace*) je nezbytné vymazat cachovaná data z databáze.

3.2.1. Cache Drupalu - databáze MySQL

Základní cache primárně integrovaná do CMS Drupal 6 umožňuje základní optimalizaci výkonu webu. Systém využívá tabulky uvedené na *obrázku č. 7: Tabulky v databázi pro uložení cache*. Ukládány jsou následující části: bloky (*cache_block*), které jsou vkládané do stránek, formuláře (*cache_form*), menu a navigace (*cache_menu*), stránky – nody (*cache_page*), views (*cahe_views*, *cache_views_data*).

Pro správnou funkci paměti cache je důležitá velikost jednotlivých tabulek. Pokud by bylo ukládáno velké množství dat a nebyla by nastavena *minimální životnost cache* (čas, po který nebude upravený obsah v cache obnoven, až po jeho uplynutí je obsah v paměti cache obnoven) a *expirace cache stránek* (maximální doba, kdy stránka zůstane v paměti cache, po jejím uplynutí musí být aktualizována), velikost tabulek by se velice rychle zvětšovala. (Trevor, 2010) To by vedlo k následnému snížení výkonu. Na uvedeném výpisu z databáze, *obrázek č. 7: Tabulky v databázi pro uložení cache*, je příklad nárůstu množství dat pro cache formulářů (*cache_form*). Tato tabulka obsahuje více jak 23 000 záznamů, 152,8 MiB dat a navíc z důvodu neprovedení optimalizace 70,6 MiB dat. Rozborem příčin a řešením se zabývá *kapitola 4.1.1 Volba a vliv typu uložiště databáze MySQL.*



cache	39	MyISAM	utf8_general_ci	1,1 MiB	123,5 KiB
cache_block	0	MyISAM	utf8_general_ci	4 KiB	-
cache_content	6 155	MyISAM	utf8_general_ci	4,9 MiB	740 B
cache_filter	7 147	MyISAM	utf8_general_ci	12,9 MiB	2,6 MiB
cache_form	23 905	MyISAM	utf8_general_ci	152,8 MiB	70,6 MiB
cache_menu	10 997	MyISAM	utf8_general_ci	10,2 MiB	-
cache_page	413	MyISAM	utf8_general_ci	20,8 MiB	13 MiB
cache_similarterms	0	MyISAM	utf8_general_ci	4 KiB	-
cache_update	3	MyISAM	utf8_general_ci	1,1 MiB	-
cache_views	17	MyISAM	utf8_general_ci	319,5 KiB	-
cache_views_data	1 144	MyISAM	utf8_general_ci	10,6 MiB	6 MiB

Obrázek č. 7: Tabulky v databázi pro uložení cache

Ukládání paměti cache do databáze MySQL je výhodné pouze pro menší webové stránky, v takových případech je toto řešení dostatečné. Přináší nárůst výkonu a nevyžaduje odborný zásah do konfigurace CMS Drupal a serverové části. Následující *kapitola 3.3 Rozšířené možnosti optimalizace výkonu* analyzuje pokročilé způsoby využití paměti cache vhodné pro nasazení při vysoké zátěži, jak anonymními uživateli, tak i přihlášenými uživateli. Příkladem může být zpravodajský web poskytující informace anonymním uživatelům – čtenářům a současně zpracovávající požadavky redaktorů, kteří obsah vytvářejí.

3.3. Rozšířené možnosti optimalizace výkonu

Při vyšším vytížení Drupalu, návštěvnost v řádu desetitisíců denně a více, a velkém množství současně přistupujících uživatelů, se stávají v předešlé kapitole uvedené základní nástroje pro optimalizaci výkonu nedostatečnými. Proto doplňkové moduly umožňují částečně zvýšit výkon. Následující tabulka srovná základní vlastnosti vybraných modulů:

Tabulka č. 1: Moduly pro cache v CMS Drupal 6 - srovnání modulů

	Druh paměti cache	Anonymní / přihlášení uživatelé
Alternative PHP Cache[25]	operační kód	oboje
Cache Static[26] **/ Boost**	soubory	anonymní uživatelé
Advanced Cache[27]	databáze	přihlášení uživatelé
Authcache[28]	databáze	oboje

Zdroj: Autor

Alternative PHP Cache (APC) je rozsáhlý otevřený framework, který umožňuje cachování a optimalizaci operačního kódu PHP. V základním nastavení je nastavena velikost této paměti cache na 32 MiB. Pro využití tohoto modulu v Drupalu 6 je nutné instalovat přes modul Cache backport.

Cache Static je modul, jehož vývoj byl ukončen a začleněn do modulu Boost. Umožňuje vytvářet statickou cache v souboru bez vyššího zatížení databáze.

Modul Boost vytvářet pokročilou cache statických stránek pro anonymní uživatele (Obrázek č. 8: Boost - rozdíl přihlášeného/nepřihlášeného uživatele). Technické řešení tohoto modulu je vhodné pro sdílený hosting, u kterého je omezení vlastní pokročilé konfigurace. (Trevor, 2010) Boost lze použít primárně pro Apache, podporovány jsou ale i Ngingx a Lighttpd. Mezi hlavní rysy patří možnost nastavení odlišné minimální životnosti paměti cache pro různé části webu a výběr stránek, které budou uloženy.

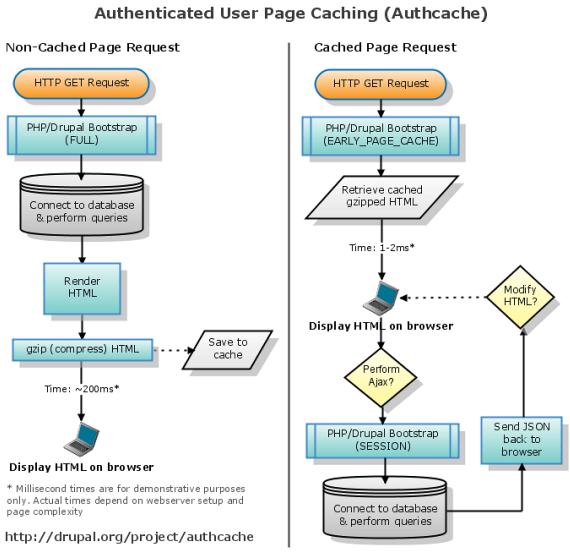
[25] http://drupal.org/project/apc
[26] http://drupal.org/project/cachestatic
[27] http://drupal.org/project/advcache
[28] https://www.drupal.org/project/authcache

Obrázek č. 8: Boost - rozdíl přihlášeného/nepřihlášeného uživatele (Carper, 2015)

Při instalaci modulu Boost pro Drupal 6 je doporučena instalace modulů *Global Redirect* a *Transliteration*. Oba moduly zajišťují korektní zpracování URL, nezbytné pro funkci modulu Boost. Řeší tím případné nekorektnosti URL (bílé znaky, kontrola URL adres aj).

Authcache (Authenticated User Page Caching) je modul, který umožňuje uložit do paměti cache stránky jak pro anonymního uživatele, tak i pro přihlášeného uživatele. Modul udržuje paměť cache renderovaných stránek. Pro každou uživatelskou roli udržuje samostatnou vlastní paměť cache. (Ellison, 2014) V prvním kroku je uživateli poskytnuta stránka uložená v paměti cache a až později je pro přihlášeného uživatele proveden další http požadavek prostřednictvím konstrukce Ajax [29] (znázorněno na *obrázku č. 9: Authcache - rozdíl pro přihlášeného/nepřihlášeného uživatele*). Využití cachování pro přihlášeného uživatele vyžaduje úpravu procesu zobrazení uživatelského obsahu. Jedná se tedy o značný zásah do celé řady dalších částí CMS Drupal.

[29] Ajax - Asynchronous JavaScript and XML technologie pro asynchronní zpracování web. stránek.



Obrázek č. 9: Authcache - rozdíl pro přihlášeného/nepřihlášeného uživatele (Ellison, 2014)

3.4. Třívrstvá architektura

Třívrstvá architektura obsahuje prezentační, aplikační a datovou vrstvu. Aplikace je poté tvořena součinností těchto vzájemně spolupracujících vrstev.

Databázová vrstva – jádro Drupalu 6 pracuje s databází MySQL nebo PostgreSQL[30]. Optimalizace na této úrovni se zaměřuje na vhodné nastavení DBMS, možnosti využití paměti cache, replikace databázového serveru a jiné. Touto částí se podrobněji zabývá *kapitola 4.1 Optimalizace databáze MySQL pro práci CMS Drupal*.

Aplikační vrstva – tvoří PHP [31] server-side skriptovací jazyk. V této vrstvě je možnost využití optimalizace frameworku Alternative PHP Cache (APC), který cachuje a optimalizuje PHP kód.

Prezentační vrstva – tato vrstva obsahuje velmi často multiplatformní webový server Apache (40% zastoupení, 30% Microsoft a 15% nginx (January 2015 Web Server Survey, 2015)). Prostor pro optimalizaci v této vrstvě je např. v odlehčení webovému serveru http akcelerátorem Varnish, který je popsán dále.

[30] PostgreSQL je multiplatformní objektově relační databázový systém, open source software.
[31] PHP, zkratka *PHP: Hypertext Preprocessor* je skriptovací programovací jazyk.

Uvedené databáze, skriptovací jazyk a webový server společně tvoří open source řešení AMP[32], doporučované vývojáři Drupalu. (System reguirements, 2014) Při optimalizaci výkonu Drupalu je nutné se zaměřit zvlášť podrobně na všechny tři oblasti.

3.5. Optimalizace – databázová vrstva

Optimalizaci databáze lze rozčlenit do dvou logických celků. Prvním pohledem je kvalita zdrojového kódu aplikace, v daném případě CMS Drupal 6. Zde je posuzován stupeň optimalizace jednotlivých dotazů ve zdrojovém kódu jádra systému a použitých volitelných modulů. Posuzován je v kontextu použitého typu uložiště databáze.

Odlišným oborem optimalizace je DBMS a jeho konfigurace v souvislosti s použitým typem uložiště a nastavením systémových proměnných. Proces optimalizace není jednorázová událost, ale průběžný proces. Je nutné stanovit postupy a procedury provádějící údržbu systému DBMS vedoucí k optimálnímu výkonu v průběhu času.

Jednou z předností CMS Drupal je modularita systému, která umožňuje vývoj celé řady doplňků, které umožňují provádět pravidelně (použitím serverové systémové služby *cron*) úkony v DBMS vedoucí k udržení stávajícího výkonu. Příkladem je níže uvedený modul DB Maintenance provádějící defragmentaci indexů. Oproti tomu možnosti zásahu do jádra CMS Drupal jsou obtížné z důvodu udržení stability systému a nejsou vývojáři doporučovány. (Drupal.org, 2014) Jakýkoliv zásah do jádra, byť by přinesl optimalizaci např. konkrétního dotazu, by tak mohl narušit stabilitu systému a v případě aktualizací (zejména kritických bezpečnostních) zvýšit riziko nekompatibility těchto aktualizací.

Takovýchto záplat (patch) existuje celá řada, a to více či méně stabilních záplat jádra systému, která přinášejí dílčí optimalizace. Jejich použití je ale více problematické než přínosné. Řešením je komplexní distribuce Pressflow, která integruje pečlivě prověřené a testované pache a úpravy celého zdrojového kódu systému se zaměřením na výkon. (Pressflow documentation, 2014)

[32] AMP, zkratka pro Apache, MySQL a PHP

3.5.1. DB Maintenance

DB Maintenance[33] je modul umožňující vybrat tabulky databáze, u kterých je provedena údržba při pravidelném spuštění skriptu cron.php. Tento skript je spouštěn automaticky, podle nastavení serveru, nebo ručně. Zvláště u typu uložiště MyISAM Drupalu 6 je důležité (z pohledu optimalizace výkonu) pravidelně provádět optimalizaci, opravu a řadění indexů (Narayan, a další, 2013). Funkci tohoto modulu lze zastoupit nastavením samotné databáze.

3.5.2. MySQL úložné enginy

Zhodnocením typů úložných enginů v souvislosti s použitím CMS Drupal se zabývá následující praktická část této práce.

3.6. Optimalizace – aplikační vrstva

Vytváření paměti cache na úrovni aplikace, tj. na vyšším stupni než databázová vrstva, představuje další krok v procesu optimalizace. Na aplikační úrovni se pracuje s mezivýsledky, načítanými daty a jejich zpracováním. Schwartz rozlišuje následující typy: *místní cache*, která je velice malá a platná pouze v rámci daného procesu (používá se např. proměnná nebo hashovaná tabulka). Dalším typem je paměť střední velikosti, řádově jednotek GB, obdoba místní cache, ale *se sdílenou pamětí*. Přístup k této paměti bývá rychlejší než k jinému používanému typu vzdálené paměti cache. Pro ukládání celých sdílených objektů typu uživatelských profilů, fragmentů HTML apod. lze využít *distribuovanou paměťovou cache*, která se může skládat z více uzlů. Oproti předešlým typům má vyšší latenci. Z tohoto důvodu je pro efektivní zpracování všech požadavků nutné jejich provedení v rámci jedné operace. Příkladem distribuované paměťové cache je memcached[34]. Posledním typem je uložení paměti *cache na disku*, využití je pro trvalé objekty (např. generované obrázky). V tomto případě lze využít pro generování cachovaných souborů zpracování chyby 404. V případě, že soubor není nalezen, je chyba

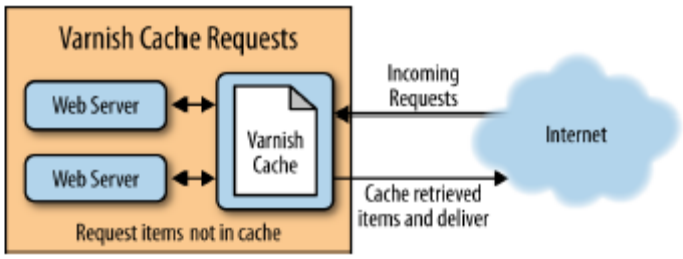
[33] https://www.drupal.org/project/db_maintenance

[34] memcached – open-source systém, který vznikl při vývoji služby LiveJournal. Umožňuje ukládání do hashovací tabulky, distribuované přes více serverů. Je využívána službami YouTube, Facebook, Wikipedia aj. (http://memcached.org/, 23. 1. 2015)

zpracována, soubor vygenerován a uložen. Následně je vygenerovaný obsah vrácen prohlížeči jako datový stream. Tímto způsobem je souborová paměť cache postupně naplněna, respektive obnovena, je-li smazána (Schwartz, a další, 2009).

3.7. Optimalizace – prezentační vrstva

V rámci optimalizace prezentační vrstvy je možné využít Varnish, který umožňuje zrychlit http požadavky, tj. tzv. „web application accelerator“. Využívá kešování http reverzních proxy (umožňuje odlehčení rozložením do více aplikačních serverů), a tím dochází ke zkrácení odezvy. Funkce aplikace Varnish je znázorněna na *obrázku č. 10: Funkce Varnish reverzní proxy*. Obecným cílem je co nejvíce obsahu udržet dostupným v operační paměti a překlenout náročné procesy pro Apache.



Obrázek č. 10: Funkce Varnish reverzní proxy (Narayan, a další, 2013)

Varnish zpracovává požadavky pro Apache, je-li to možné, využije paměť cache. Varnish je instalován mezi uživatele a Apache. Tímto způsobem je zpřístupněn obsah pro anonymní uživatele ve formě statických souborů. Dojde k odlehčení anonymních požadavků pro Apache a získání více systémových prostředků pro přihlášené uživatele a jiné požadavky. V tomto případě je Apache konfigurován na nestandardním portu 8080.

3.8. Pressflow

Pressflow [35] je speciální distribuce Drupalu optimalizovaná pro velmi vysoký výkon. Využívají ho například webové portály *Stanford University, eTech media*, *data.gov.uk*, *Yale University*, *NBC-Universal*, *The Grammy Awards*, *Central European University* a jiné. Vývoj této distribuce vede vývojová a poradenská společnost Four

[35] http://pressflow.org/

Kitchens[36], která se specializuje na open-source řešení a je zejména spojená se systémem Drupal.

Kód jádra systému Drupal je neměnný, respektive není doporučeno do části jádra zasahovat z důvodu zajištění stability celého systému. Z tohoto důvodu je prostor pro optimalizaci značně zúžený. Toto omezení překlenuje právě Pressflow, jehož zdrojový kód je upravený. I přes celou řadu změn je Pressflow s Drupalem v příslušné verzi API ekvivalentní (pokud nejsou volané privátní funkce) a také schéma ekvivalentní pro databázi. (Pressflow documentation, 2014) Migrace mezi oběma systémy je možná obousměrně.

Na *tabulce č. 2: Srovnání Pressflow a Drupalu 6, 7* je názorné srovnání vlastností, které jsou důležité pro výkon CMS Drupal. Drupal 6 je výkonově zcela nevyhovující (nepodporuje replikaci, external page cache a další sledované parametry). Oproti tomu Pressflow 6 je ve sledovaných parametrech téměř srovnatelné s Drupalem 7, navíc obsahuje multi-layer proxy support.

Tabulka č. 2: Srovnání Pressflow a Drupalu 6, 7

	Drupal 6	**Pressflow 6**	**Drupal 7**
Odebrání zámků	ano	ano	ano
Replikace databáze		ano	ano
External page cache		ano	ano
Fast path alias detection		ano	ano
First path argument whitelist		ano	ano
Native JSON encoding		ano	ano
Multi-layer proxy support		ano	
Lazy session creation		ano	ano
Path caching		ano*	ano

Zdroj: (Pressflow documentation, 2014), (Comparison Pressflow, Drupal, 2015) ** probíhá vývoj*

Replikace databáze – přidává podporu pro nastavení replikace databáze. V souboru settings.php je přidána možnost nastavení *$db_slave_url* pro databázový podřízený *slave-server* a *$db_url* pro databázový výchozí řídící tzv. *master-server*. Podrobněji je tato problematika analyzována v kapitole 4.1.3 Replikace databáze MySQL a v následující kapitole 4.2.1 Replikace databáze s Pressflow 6.

[36] Four Kitchens je poradenská a vývojová společnost sídlící ve městě Austin, Texas.

External page cache – umožňuje využít technologii pro cachování stránek bez použití PHP a databáze, např. Varnish.

Srovnáním CMS Drupal 6 a Pressflow 6 se zabývá praktická část této práce, zejména využitím profilace systému Drupal.

3.9. Jazyk SQL

Vývoj jazyka SQL započal už v roce 1974 na pracovišti IBM Santa Teresa v San Jose v Kalifornii. Původní projekt byl pojmenován System/R. Funkce tohoto systému ověřila použití relačního modelu. Ve stejném roce byl publikován článek, autorů System/R, Donalda D. Chamberlina a R. F. Boyce *Sequel: A Structured English Query Language.* (Chamberlin, et al., 2015) Nový jazyk SEQUEL, který byl implementován do databáze SEQUEL-XRM, umožňoval pokládat dotazy nad relační databází srozumitelně a podobně jako v anglickém jazyce. Vývoj pokračoval mezi lety 1976 a 1977, kdy probíhal výzkum jazyka SEQUEL, respektive SEQUEL/2. Jazyk byl přejmenován z právních důvodů na SQL (strukturovaný dotazový jazyk - Structured Query Language). (Pekárek, 2012)

V průběhu 70. let bylo postupně zřejmé, že firma IBM uvažuje o komerčním využití relační databázové technologie. Tato nová technologie byla diskutována na odborných seminářích a časopisech. To vedlo skupinu odborníků k založení nové společnosti Relational Software Inc.[37], která vytvořila RDBMS[38] – systém správy relačních databází Oracle. Oracle byl v roce 1979 uveden na trh, tedy o dva roky dříve než produkt společnosti IBM a následně byla společnost přejmenována na Oracle Corporation.

Výzkum relační databáze probíhal i na univerzitě v Berkeley pod vedením M. Stonebrakera a E. Wonga. Zde vznikl prototyp relační databáze INGRES s databázovým jazykem QUEL (dotazový jazyk – Query Language), který byl ve srovnání s SQL více strukturovaný. Pod pozdějším tlakem k používání SQL, jazyk SQL se stal více rozšířeným, došlo k přechodu systému INGRES v systém správy relačních databází SQL. (Pekárek, 2012), (Gilmore, 2007) Následně v roce 1981 byl firmou Relational Technology uveden komerční produkt INGRES.

[37] Relational Software Inc. Byla založena v roce 1977, Menlo Park v Kalifornii.
[38] RDBMS – Relational Database Management System

Jazyk SQL se skládá ze tří částí: jazyk pro definici dat – DDL (Data Definition Language), jazyk pro manipulaci s daty – DML (Data Manipulation Language), tj. příkazy pro vkládání, mazání a jazyk pro definici přístupových práv k datům DCL (Data Control Language).

3.10. Databáze MySQL

Cílem této práce je optimalizace databáze pro CMS Drupal. Protože ve verzi Drupalu 6 jsou podporovány databáze PostgreSQL a právě MySQL, která je využita i v instalaci Drupalu pro redakční systém univerzitních novin iZUN.eu, je tato práce právě zaměřena na tuto databázi.

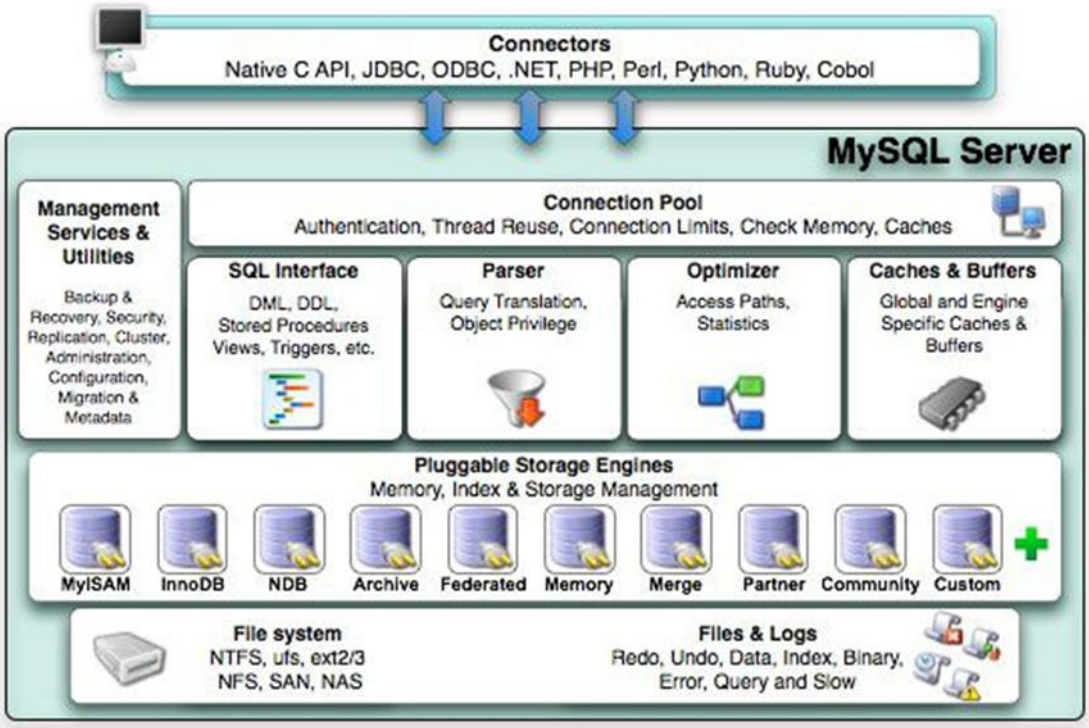
3.10.1. Úvod - historie

Společnost MySQL AB [39] vznikla ve Švédsku v roce 1995. Od roku 2000 byla databáze MySQL dostupná i pod licencí GPL. V roce 2008 došlo ke změně vlastníka Sun Microsystems a o rok později MySQL přešlo pod Oracle. (Pekárek, 2012 str. 13)

3.10.2. Architektura databázového serveru

Architektura MySQL je tvořena třemi vrstvami (*obrázek č. 11: Schéma MySQL serveru*). V první vrstvě jsou služby, které umožňují komunikaci klient/server, autentizaci a vznik vláken. Druhá vrstva obstarává parsing SQL. Rozebráním dotazu vytvoří stromovou strukturu parse-tree. Určuje se pořadí přístupu k daným tabulkám, indexům atd. Tato vrstva obsahuje také cache dotazů. Třetí vrstva obsahuje úložné enginy. Tyto úložné enginy komunikují se serverem prostřednictvím API jednotlivých úložných enginů. Skrze toto API je umožněna komunikace pro odlišné úložné enginy. (Pekárek, 2012 str. 14), (Schwartz, a další, 2009)

[39] Společnost založili: M. Widenius, D. Axmark a A. Larsson.



Obrázek č. 11: Schéma MySQL serveru (Valenta, 2014)

3.10.3. Základní typy uložišť

MySQL ukládá každou databázi zvlášť do podadresáře v hlavním datovém adresáři. V adresáři databáze jsou uloženy definice jednotlivých tabulek ve formátu *název_tabulky.frm*[40], definice jsou uloženy také v hlavní databázi MySQL. Každý úložný engine už ale data a indexy ukládá sám podle své vlastní specifikace. MySQL primárně integruje uložiště InnoDB, MyISAM, MEMORY[41], Merge, solidDB, Archive, CSV, Federated, NDB Cluster a Maria.

MyISAM

Uložiště MyISAM je nástupcem formátu ISAM (Indexed Sequential Access Method) od společnosti IBM. Každá tabulka je tvořena hlavním souborem obsahujícím sekvenčně uložené záznamy pevné délky a k tomu alespoň jedním souborem indexů.

[40] Rozlišování velikosti písmen závisí na operačním systému UNIX/Windows
[41] do verze MySQL 5.0 označované HEAP (Schwartz, a další, 2009)

Tabulky MyISAM lze definovat se statickými řádky, nebo s dynamickými řádky. Maximální velikost tabulky je determinována maximální velikostí souboru, který umožňuje vytvořit operační systém a velikostí volného místa. Třetím typem tabulek MyISAM je tzv. komprimovaný typ, který lze využít ve specifických aplikacích, které data z databáze pouze čtou.

Při zápisu MyISAM uzamyká celé tabulky, díky tzv. souběžnému vkládání je možné vkládat nové řádky i v průběhu vykonávání výběrového dotazu nad danou tabulkou.

InnoDB

Typ uložiště InnoDB je v MySQL od verze 3.23.24, byl vytvořen a dále rozvíjen společností Innobase. Umožňuje širokou správu nad operacemi prováděnými s databází. Je vhodné z pohledu bezpečnosti, konzistence dat, také z pohledu výkonu a zamykání na úrovni řádků (tzv. multi processing na jedné tabulce). Prováděné operace lze spouštět jako transakce. „*To umožňuje zvýšení bezpečnosti. Operace se může skládat z logicky souvisejících příkazů. Ty se buď provedou všechny, anebo žádný. Podle standardu ANSI-SQL/92 se transakce spouští v izolačních úrovních READ UNCOMITTED, READ COMMITED, REPEATABLE READ, SERIALIZABLE. Při transakci dochází k zamykání na úrovni záznamů, pro ostatní uživatele nedochází k omezování přístupu.*“ (Pekárek, 2012 str. 15)

InnoDB ovladač zajišťuje referenční integritu. Neměla by nastat situace, kdy je odkazováno na neexistující záznam jiné tabulky. (Gilmore, 2007), (Schwartz, a další, 2009)

MEMORY

Uložení dat v typu uložiště MEMORY je realizováno pouze v paměti RAM. Tento typ je velice rychlý a slouží například ke komunikaci mezi procesy. Během praktického měření rychlosti lze dosáhnout až o 50% rychlejšího přístupu k datům než u typu uložiště InnoDB[42].

[42] „[...] vytvořili jsme tři identické kopie tabulky message_buffer, které se liší pouze druhem úložiště. Vybrali jsme možnosti úložiště MYISAM, INNODB a MEMORY. Po vytvoření tabulek jsme do všech načetli velké bloky náhodných dat. Tyto operace byly u tabulek MEMORY v průměru o 33% rychlejší než u MYISAM a o 50% rychlejší než u INNODB.“ (Schneiderx, 2009 str. 85)

NDB Cluster

MySQL integrovalo NDB Cluster pro operace vyžadující vysokou rychlost a rozložení zátěže. Veškerá data jsou udržována v operační paměti a optimalizována pro vyhledávání podle primárního klíče. Podle B. Schwartze je pojetí NDB Clusteru velice odlišné ve srovnání například s clusterem Oracle (Schwartz, a další, 2009).

solidDB

SolidDB je transakční engine využívající souběžného zpracování s více verzemi (MVCC[43]). MVCC je využíváno i jinými databázovými systémy, např. Oracle, PostgreSQL. Podporovány jsou oba režimy souběžnosti, jak pesimistický, tak i optimistický režim. Svými vlastnostmi a podporou cizích klíčů se solidDB podobá InnoDB, navíc umožňuje provádět zálohování online (Schwartz, a další, 2009).

[43] MVCC – Multi Version Concurrency Control.

4. Praktická část

Optimalizace databáze je jedna z částí komplexní problematiky optimalizace celého systému – CMS Drupal a souvisejících částí. Pomocí testování výkonu v plném rozsahu, tedy se zapojením všech komponent (databáze, webový server) a profilace aplikace, lze objektivně zhodnotit výkon celého systému. Analýzou získaných výsledků lze následně stanovit tzv. úzká hrdla systému a na ně se podrobněji zaměřit. Testy výkonnosti slouží ke zhodnocení systému jako celku, určení kapacity systému, reakce na různé druhy vstupních dat, anebo určení vlivu různých změn. Profilace analyzuje jednotlivá místa aplikace, čas strávený dílčími kroky a množství spotřebovaných systémových zdrojů.

4.1. Optimalizace databáze MySQL pro práci CMS Drupal

Optimalizaci databáze lze rozdělit do dvou úrovní. Na úroveň návrhu databáze, její struktury a také volby vhodných typů uložišť. Druhou úrovní je optimalizace v průběhu využívání databáze, respektive její průběžná údržba.

4.1.1. Volba a vliv typu uložiště databáze MySQL

Využití databáze systémem Drupal 6 je omezeno doporučenou konfigurací databáze na použití úložného enginu MyISAM. Podle analýzy v teoretické části práce se jedná o typ uložiště, který není nejlepší pro všechny činnosti systému.

MEMORY

MySQL obsahuje v základní konfiguraci úložný engine MEMORY, který ukládá obsah tabulek přímo do paměti RAM. Tímto způsobem je dosahováno mnohem vyšší rychlosti a kratší odezvy než u klasického způsobu uložení dat na pevný disk.

Možnost využití typu MEMORY by bylo vhodné pro obsah, který je relativně snadno obnovitelný, a to v případě výpadku serveru a následné ztráty dat. Předpoklady pro tento typ obsahu splňuje paměť cache. Obecně byla struktura části databáze pracující s pamětí cache Drupalu 6 popsána v kapitole 3.2.1 Cache Drupalu - databáze MySQL.

Název	Typ	Porovnávání	Vlastnosti	Nulový	Výchozí	Další	Operace
cid	varchar(255)	utf8_general_ci		Ne			Změnit Odstranit Primární Unikátní Klíč
data	longblob			Ano	*NULL*		Změnit Odstranit Primární Unikátní Klíč
expire	int(11)			Ne	0		Změnit Odstranit Primární Unikátní Klíč
created	int(11)			Ne	0		Změnit Odstranit Primární Unikátní Klíč
headers	text	utf8_general_ci		Ano	*NULL*		Změnit Odstranit Primární Unikátní Klíč
serialized	smallint(6)			Ne	0		Změnit Odstranit Primární Unikátní Klíč

Obrázek č. 12: tabulka cache - struktura

Podrobnější rozbor struktury tabulky *cache* (*obrázek č. 12: tabulka cache - struktura*) obsahuje datový typ *longblob* pro sloupec *data* obsahující hlavní data a datový typ *text* pro sloupec *headers*. Oba jmenované datové typy neexistují u úložného enginu MEMORY. Možnost převodu na jiný datový typ nelze provést z důvodu omezení maximální velikosti běžných datových typů (*VARCHAR, CHAR, ...*), která je menší než u *longblob* a *text*. Dále MySQL umožňuje uložení maximální velikost řádku 65 535 bytes (2015). Tato omezení jsou určující pro závěry vyplývající z této analýzy uložených dat CMS Drupal 6.

←T→	cid	data	expire	created	headers	serialized
Upravit Kopírovat Odstranit	schema	[BLOB - 154,1 KiB]	0	1425851359		1
Upravit Kopírovat Odstranit	imagecache:presets	[BLOB - 3,8 KiB]	0	1425851360		1
Upravit Kopírovat Odstranit	theme_registry:izun2012	[BLOB - 175,5 KiB]	0	1425851361		1
Upravit Kopírovat Odstranit	variables	[BLOB - 71,7 KiB]	0	1425851445		1
Upravit Kopírovat Odstranit	locale:cs	[BLOB - 28,1 KiB]	0	1425853951		1

Obrázek č. 13: tabulka cache - data

Příkladem využití tabulek paměti cache ve vzorové instalaci Drupalu 6 pro redakční systém novin iZUN.eu je samotná tabulka *cache* (*obrázek č. 13: tabulka cache - data*), zvláště pak sloupec *data*, který zabírá velikost v řádu KiB a až desítek KiB. Datový typ pro tento sloupec není možné převést na jiný datový typ, podporovaný enginem MEMORY.

Přechod na typ uložiště MEMORY pro základní cache Drupalu není z důvodu struktury dat (použitých datových typů) a velikosti ukládaných dat možný.

4.1.2. Optimalizace uložiště MyISAM

Optimalizace databáze na úrovni jednotlivých uložišť je rozdělena do dvou částí podle přístupu a typu použitých nástrojů, respektive programů. První skupinu tvoří

nástroje v prostředí serveru umožňující provádět jednotlivé operace, obecně přístupné z terminálu. Tento způsob je vhodný při profesionální správě databázového serveru umožňující operace automatizovat. Oproti tomu druhá skupina nástrojů zpřístupňuje správu databáze skrze jiná uživatelská rozhraní (např. webové rozhraní *phpMyAdmin*), nebo skrze samotné moduly a rozšíření CMS Drupal.

Optimalizace v prostředí terminálu – program myisamchk

Drupal 6 využívá primárně v databázi MySQL úložný engine MyISAM. Každá tabulka se skládá ze tří souborů (**.frm* formátovací soubor obsahující datovou strukturu, **.myd* obsahující data a **.myi* obsahující data o klíčích a křížových odkazech). Pro získání podrobných statistických informací, kontrolu, optimalizaci a opravy lze využít program *myisamchk*.

Při použití *myisamchk* je nutné zamezit přístupu uživatelů k databázi, ukončit databázový server, nebo případně uložit data z mezipaměti MySQL. Uložení dat z mezipaměti lze provést příkazem klienta *mysqladmin:*

```
shell> mysqladmin flush-tables
```

Příkazy programu *myisamchk* lze spouštět v adresáři příslušné databáze se kterou se pracuje, nebo obecně v hlavním databázovém adresáři. (2015) Příkazy lze také specifikovat pro jednu, nebo všechny tabulky v dané databázi zástupným znakem *:

```
shell> myisamchk /cesta/k/databazovemu_adresari/*.MYI
```

Nebo lze zadat příkazy pro všechny databáze v *data_adresari* MySQL:

```
shell> myisamchk /cesta/k/data_adresari/*.MYI
```

Základní příkaz pro kontrolu a případnou opravu všech tabulek ve všech databázích vzorové instalace iZUN.eu lze provést následujícím příkazem:

```
1. shell> myisamchk --silent --force --fast --update-state \
2.           --key_buffer_size=64M --sort_buffer_size=64M \
3.           --read_buffer_size=1M --write_buffer_size=1M \
4.           /var/mysql/data/izun/*.MYI
```

Použití programu *myisamchk* není vždy možné. Buď to neumožňuje provozovatel serveru (případy jednodušších hostingů na principu služby), anebo není možné odpojit všechny klienty od databáze. V daném případě nasazení serveru v prostředí, kde není možné odpojit všechny klienty od databáze MySQL lze využít SQL dotaz:

```
CHECK TABLE název_tab... [možnosti]
```

Tento způsob lze využít i pro *pohledy* a kontrolu jejich aktuálnosti. Opravit poškozené tabulky lze také SQL dotazem:

```
REPAIR [NO_WRITE_TO_BINLOG | LOCAL] TABLE
   Název_tab [, název_tab] ...
   [QUICK] [EXTENDED] [USE_FRM]
```

Uvedený příkaz je obdobou programu *myisamchk* s parametrem –*recover.*

Optimalizace využitím modulu CMS Drupal - DB Maintenance

Ne vždy je možné na databázovém serveru spouštět programy pro optimalizaci. Často u méně náročných provozovatelů hostingových služeb je jedinou možností optimalizace databáze použitím SQL dotazů. Tento způsob využívá i modul *DB Maintenance*. V pravidelných intervalech, spuštěných skriptem cron.php [44] , jsou prováděny optimali-zační SQL dotazy nad vybranými tabulkami. Na uvedené části kódu souboru *db_maintenance.module* jsou uvedeny SQL dotazy na řádku 146. pro databázi MySQL, respektive na řádku 149. pro databázi PostgreSQL. Pro kontrolu a optimalizaci tabulek databáze MySQL se používá dotaz *OPTIMIZE TABLE název_tab* a v případě druhé, pro Drupal nejčastěji využívané databáze PostgreSQL, je používán dotaz *VACUUM ANALYZE název_tab.* V případě databáze MySQL se jedná o odlišný dotaz, než byl uveden v předešlé části jako alternativa k využití programu *shell> myisamchk.* Činnost obou dotazů ale vede ke stejným výsledkům.

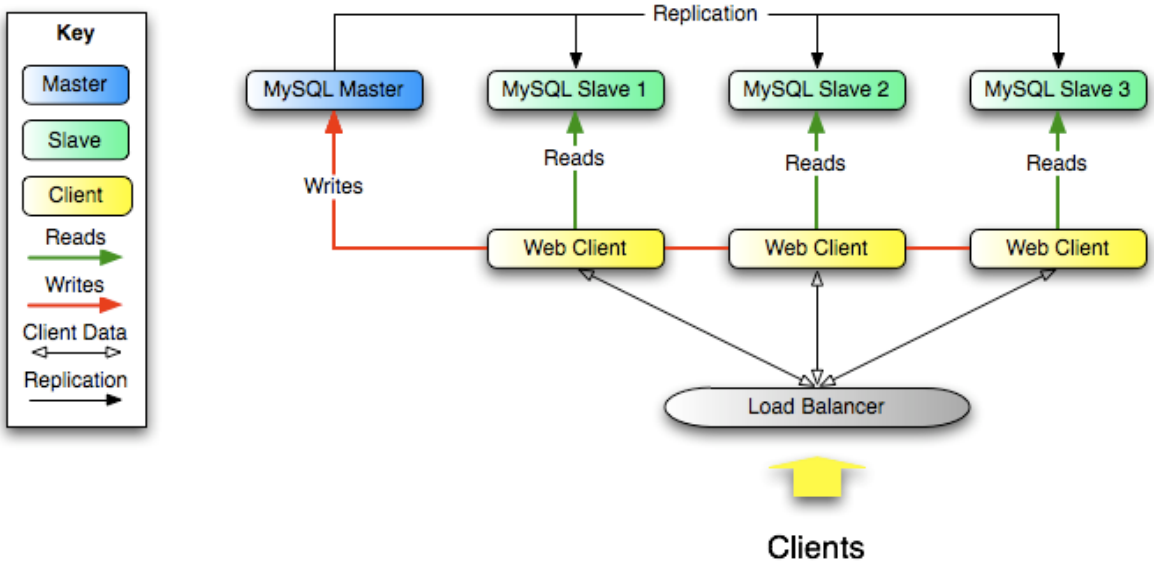
```
140. … while (list(, $table_name) = each($config_tables)) {
141.// Set the database to query.
142.$previous = db_set_active($db);
143.$table_check_name = $is_prefixed ? str_replace($db_prefix, '', $table_name) : $table_name;
144.    if (db_table_exists($table_check_name)) {
145.            if (_db_maintenance_determine_software() == 'mysql') {
146.                    db_query('OPTIMIZE TABLE %s', $table_name);
147.            }
148.            elseif (_db_maintenance_determine_software() == 'pgsql') {
149.                    db_query('VACUUM ANALYZE %s', $table_name);
150.            }
151.    }
152.    else { …
```

4.1.3. Replikace databáze MySQL

Replikaci databáze MySQL lze využít pro distribuci dat a předejít tím nárůstu latence a zlepšit geografické rozčlenění. Replikace může být použita i pro analytické nástroje a k rozložení zátěže mezi více serverů. Tento způsob umožňuje

[44] cron.php – pravidelně spouštěný skript.

zvýšit bezpečnost, pokud se jedná o specifický způsob vytváření záloh. Respektive vytváření záložního, stále aktuálního serveru, který může být vždy připraven převzít zátěž primárního serveru (podrobněji v topologii replikace). MySQL umožňuje využít odlišné režimy replikace databáze, primárně: asynchronní a semisynchronní režim. V rámci replikace lze jednu databázi rozdělit, respektive její tabulky do různých replikačních skupin. Uvedený *obrázek č. 14: Replikace pro zvýšení výkonu, škálovatelnost* znázorňuje jednu z výše jmenovaných možností využití replikace pro rozložení zátěže mezi více podřízených serverů.



Obrázek č. 14: Replikace pro zvýšení výkonu, škálovatelnost (MySQL 5.6 Reference Manual, 2015)

V případě asynchronní master-slave replikace lze využít podřízené servery pouze pro čtení (příklad rozdělení směrů pro *čtení* a *zápis* na *obrázku č. 14 Replikace pro zvýšení výkonu, škálovatelnost*). Zápis do podřízených serverů by jinak vedl k nekonzistenci dat. Omezení replikace je v tzv. replikačním zpoždění u podřízených serverů, které může v některých případech vést ke ztrátě dat. Tento jev je způsoben procesem, jakým je replikace prováděna. V konfiguraci master-slave řídící server ukládá prováděné operace do binárního logu. Ukládání probíhá ve třech odlišných režimech, které je možné zvolit v konfiguračním souboru *my.cnf* (MySQL 5.6 Reference Manual, 2015).

- STATEMENT režim ukládá do binárního logu přímo SQL příkazy.
- ROW režim ukládá změněné řádky.
- MIXED režim kombinuje oba předešlé režimy, kdy v některých případech volí režim ROW, vyžaduje-li to zachování konzistence dat.

Podle zvoleného režimu vzniká na *master-serveru* binární log událostí, ke kterému asynchronně přistupují *slave-servery*. V asynchronním režimu neexistuje zpětná kontrola pro *master-server,* zda provedené transakce byly přeneseny na *slave-servery.* Řešením této problematiky je od verze MySQL 5.6 Semisynchronní replikace.

V režimu semisynchronní replikace je provedena transakce úspěšně jen v případě, že se *master-serveru* podaří spojit a informovat *slave-server* o prováděné změně, nebo dojde k překročení nastaveného času pro spojení obou serverů.

V rámci MySQL replikace lze rozlišit základní topologie, které ale mají společnou vlastnost, a to že *slave-server* - replika může mít pouze jeden *master-server*. Nelze tedy použít tzv. topologii *multi-master* replikace.

- Master – více replik, základní topologii master a replika, lze rozšířit o více replik.
- Master – master (aktivní - aktivní), tzv. obousměrná replikace v režimu obou aktivních serverů má využití v případech geografického oddělení.
- Master – master (aktivní – pasivní), topologie vhodná pro přepínání role aktivního a pasivního serveru a tím pomoci odstranit případné vypadlé servery, nebo provádět údržbové a optimalizační práce bez výpadků.
- Master s replikami – master s replikami, umožňuje v geograficky distribuované topologii odstranit kritické místo výpadku.
- Master – distribuční master server a repliky, umožňuje prostřednictvím distribučního master-serveru odlehčit od zátěže způsobované replikami. V tomto případě je vedeno pro každou repliku vlákno na distribučním serveru a nikoliv na master serveru. Distribuční master-server využívá úložný engine Blackhole.
- Prstenec – tři a více serverů master, je velice zranitelná topologie, kde může vzniknout nekonečný cyklus (Schwartz, a další, 2009).

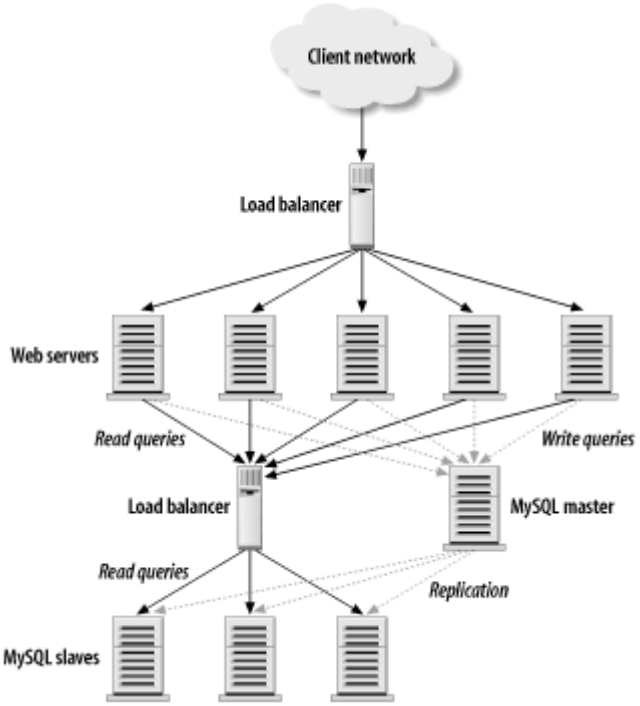
- Strom umožnuje odlehčit zátěž, která je kladena na master-server velkým množstvím replik. Obdobně jako topologie s distribučním master serverem.

V průběhu ladění replikace se můžeme setkat s různými chybami. Pro méně závažné lze zvolit tzv. přeskakování chyb, např. pro opakované vkládání řádku - duplicate entry s kódem 1062 lze nastavit na slave-serveru výjimku. V konfiguračním souboru *my.cnf* by bylo nastaveno *slave-skip-errors =1062*. Obdobně lze postupovat i u jiných chyb, je však nutné zvážit příčiny vyvolávající tyto chyby, aby nedocházelo k nekonzistenci dat mezi *master* a *slave-serverem*.

Drupal 8 a 7 umožňuje nastavit *slave-server* pro SQL dotazy čtení a *master-server* pro ostatní dotazy. (Narayan, a další, 2013) Podpora replikace je také jednou z hlavních vlastností Pressflow 6, lze ji tedy prostřednictvím Pressflow 6 implementovat pro starší instalaci Drupalu 6.

Rozložení zátěže – load balancing

Rozložení zátěže mezi skupinu serverů je prováděno specifickým zařízením, tzv. Load balancerem, který je umístěn mezi webovým serverem a replikovanými databázovými servery. Je uveden na *obrázku č. 15: Komplexní případ využití Load balancingu a replikace MySQL databáze.* Znázorněné schéma rozšiřuje předešlý případ replikace o kolekci webových serverů, jejichž vytížení je řízeno také load balancerem. Tento případ, kdy je load balancer využit na dvou úrovních je velice komplexní, náročný na správu a typický pro velmi rozsáhlé webové aplikace (Schwartz, a další, 2009).



Obrázek č. 15: Komplexní případ využití Load balancingu a replikace MySQL databáze (Schwartz, a další, 2009)

Rozložení zátěže umožňuje realizovat čtyři základní kritéria (Schwartz, a další, 2009). První oblastí je *škálovatelnost*, kdy je možné měnit množství zapojených slave-serverů a tím reagovat na změnu zátěže. Z pohledu *efektivity* lze vhodně rozlišit různě výkonné servery a podle toho mezi ně směrovat požadavky. Robustně navržená replikace zvyšuje *dostupnost*. Celé řešení pod load balancerem vystupuje *transparentně* jako jediný virtuální databázový server. V případě využívání transakcí by měl load balancer zajistit *konzistentnost* a směřovat související požadavky na daný server.

Pro rozložení zátěže nemusí vždy sloužit hardwarové zařízení a speciální software, které je umístěné mezi MySQL replikací a webovým serverem. V určitých případech je řízení rozložení zátěže na aplikaci, která určuje, k jakým serverům se může připojit. V takovém případě aplikace ošetřuje stáří dat jednotlivých replikačních serverů vzniklé asynchronním režimem replikace. Určitá část požadavků vyžaduje aktuální data, například právě editovaný článek, který je uložen a znovu načten. Podle nastavení řízení replikací by byl článek dostupný na replikačních serverech řádově s několikavteřinovým a větším zpožděním. Mezitím musí aplikace směrovat čtecí dotazy na stejný server jako zapisovací, tedy na *master-server* databázi.

Obecně lze přístup k dělení na zápisy a čtení shrnout do uvedených kategorií. Nejméně efektivní a nejjednodušší řešení je směrování na repliky pouze ty čtecí dotazy, u kterých je přípustné určité zastarání dat. V praxi se podle (Schwartz, a další, 2009) v tomto případě repliky příliš často nevyužívají.

Pokročilejším přístupem je dělení podle relace, kdy si aplikace uchová údaj o tom, zda uživatel změnil data. Pro tohoto uživatele následně bude směřovat čtecí dotazy na *master-server* databázi, aby zajistil, že uživatel uvidí aktuálně svoji změnu. Rozšířením tohoto přístupu je možnost měření doby od poslední úpravy, při znalosti maximálního zpoždění replik, po jejímž uplynutí lze bezpečně číst aktualizovaná data i z replik.

Pokud jsou využívány objekty, lze zaznamenat číslo verze objektu, respektive časový zámek z repliky a následně se rozhodnout, zda takováto data nejsou zastaralá. A podle toho se rozhodnout zda číst data ze *slave-serveru,* anebo z *master-serveru.* V tomto případě narůstá režie při zjišťování verze objektu, respektive časového zámku.

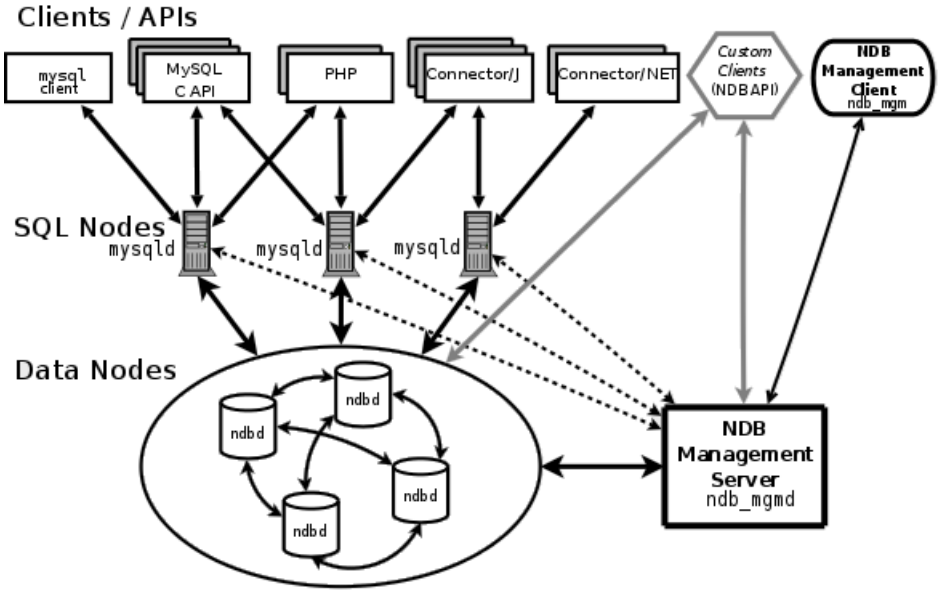
Posledním přístupem je kombinace posledních dvou, kdy jsou aplikací zaznamenávány souřadnice logu po provedení transakce (SHOW MASTER STATUS). Při dotazu na replikační server je porovnán záznam souřadnic repliky (SHOW SLAVE STATUS) a na základě toho je rozhodnuto, zda data repliky jsou zastaralá, či nikoliv (Schwartz, a další, 2009), (MySQL 5.6 Reference Manual, 2015).

Rozložení zátěže řízené aplikací přináší při dodržení kvalitního návrhu efektivitu. Řešení je ale ve srovnání s pouhou škálovatelností náročnější.

4.1.4. MySQL Cluster

Replikace popsaná v předešlé kapitole nese omezení asynchronicity, vždy je možné zapisovat pouze do jednoho master-serveru, s čímž souvisí riziko jediného kritického místa selhání. Toto riziko lze do jisté míry eliminovat použitím popsané konfigurace, kdy je například zátěž převedena na slave-server, který přejímá roli master-serveru. Jiným způsobem přinášejícím řešení této problematiky, dosažení maximální dostupnosti služeb a redundance jednotlivých kritických bodů, je MySQL Cluster (NDB – Network Database).

MySQL Cluster je vhodné použít pro databáze s vysokým provozem. Systém uchovává veškerá data a indexy v operační paměti RAM serveru. Cluster je relativně snadno rozšířitelný za provozu, lze měnit, nebo přidávat servery za plného provozu. Jde také rozdělit zpracování jediného dotazu mezi více serverů v clusteru, a tím zvýšit výkon celé databáze (MySQL 5.6 Reference Manual, 2015).

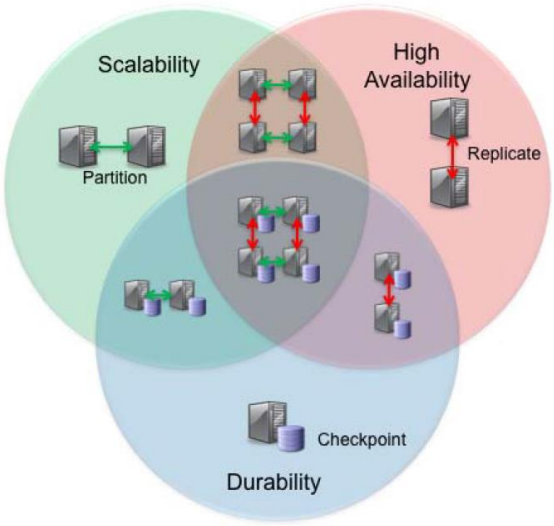


Obrázek č. 16: MySQL Cluster - vztah jednotlivých komponent (MySQL 5.6 Reference Manual, 2015)

Cluster MySQL se skládá z částí zobrazených na *obrázku č. 16: MySQL Cluster - vztah jednotlivých komponent*. Jádro tvoří množina serverů (označovaná jako *host*). Každý server zpracovává jeden, nebo víc procesů (označované jako *ndb d,* respektive *Data Nodes* na uvedené grafice). Každý nod obsahuje MySQL server s nainstalovaným úložným enginem NDB, který ukládá data do paměti RAM. Klienti (*Clients/APIs*) přistupují ke clusteru skrze *SQL Nodes*, tedy k uzlům, které se chovají obdobně jako MySQL servery, přičemž se na pozadí připojují k *Data Nodes.* Posledním prvkem je *NDB Management Server (ndb_mgmd)*, který udržuje konfiguraci pro zavádění nových uzlů, nebo při jejich spuštění a určuje, který uzel se bude jak chovat v případě síťové poruchy. Skrze tento server lze také provádět online zálohu celého clusteru.

MySQL Cluster vyžaduje minimálně tři servery a specifický hardware (Davies, a další, 2006), respektive paměť RAM, než předešlé řešení, replikace.

NDB Cluster je svojí architekturou vhodný pro aplikace, které mají relativně méně dat a obsahují jednodušší dotazy, neboť u dotazů obsahujících spojení tabulek, tedy složitějších dotazů, nedosahuje MySQL Cluster požadovaného výkonu. Takové dotazy totiž vyžadují meziuzlovou komunikaci, která je časově neefektivní (Schwartz, a další, 2009). NDB Cluster je vhodný pouze pro dotazy indexového vyhledávání podle hodnot klíče v dané tabulce.



Obrázek č. 17: Možnosti konfigurace Clusteru kombinací škálovatelnosti, vysoké dostupnosti a odolnosti (Whitepaper, 2015)

MySQL Cluster není vhodný pro CMS Drupal jednak z důvodu použití datových typů text, blob aj., které nejsou efektivní pro úložný engin MySQL Clustru a také z důvodu značné složitosti dotazů (jejich spojování přes více tabulek), které nepřináší očekávaný nárůst výkonu.

4.1.5. Optimalizace dotazů

MySQL zpracovává dotazy ve formě spojení obsahující vnořené cykly. Přesto ne všechny konstrukce dotazů dokáže optimalizátor integrovaný v MySQL efektivně zpracovat a optimalizovat sám. Následující výběr uvádí některé případy, kdy je možné v aplikaci zapsat dotazy efektivněji.

Při ladění jednotlivých dotazů a pro zjištění jakým způsobem MySQL optimalizuje dotazy je použit příkaz EXPLAIN. Výstup lze pro další zpracování formátovat do JSON.

```
EXPLAIN format=json SELECT * FROM tabulka WHERE 1
```

Dotazy LIMIT a OFFSET

Dotazy typu LIMIT a OFFSET jsou nejčastěji využívány v redakčních systémech pro tzv. stránkování, tedy daný počet vypsaných údajů na jednotlivých stránkách. Předpokladem je u takovéto tabulky, u níž bude prováděn tento typ dotazu, aby byl vytvořen index podporující řazení. Předejde se tak tomu, aby databázový server, např. MySQL prováděl přílišné množství operací filesort (Schwartz, a další, 2009).

```
SELECT tabulka_id, text1, text2  FROM tabulka ORDER BY nazev LIMIT 2000, 10;
```

Výše popsaný dotaz lze optimalizovat použitím indexu, který obsahuje malé množství dat. Nalezené řádky indexu spojíme s řádky celé tabulky a získáme požadované sloupce text1 a text2.

```
SELECT tabulka.tabulka_id, tabulka.text1, tabulka.text2 FROM tabulka tabulka
       INNER JOIN (SELECT tabulka_id FROM tabulka
              ORDER BY nazev LIMIT 2000, 10) AS lim USING(tabulka_id);
```

Profilaci MySQL lze využít pro získání uceleného přehledu o prováděných dotazech, jejich četnosti, trvání a konkrétních datech, ke kterým databáze MySQL přistupuje. Nejčastějším způsobem je zaznamenávání dotazů do obecného logu pro všechny dotazy ještě než se začnou vykonávat a log pomalých dotazů pro dotazy, které byly úspěšně dokončeny, ale trvaly déle než stanovené minimum. Následující příklad nastaví log pomalých dotazů trvajících déle než 1 sekundu v konfiguračním souboru MySQL *my.cfg*:

```
[mysqld]
slow-query-log = 1
slow-query-log-file = /var/log/<nazev_logu>
long_query_time = 1
```

Uvedený příklad konfigurace *my.cfg* vyžaduje restart databázového serveru, pro případ spuštění záznamu logu bez nutnosti restartu lze zapnout log z prostředí konzole *mysql*:

```
shell> mysql -e 'SET GLOBAL slow_query_log=1;'
shell> mysql -e 'SET GLOBAL slow_query_log_file="/var/log/<nazev_logu>";'
shell> mysql -e 'SET GLOBAL long_query_time=1;'
```

Vyhodnocením získaného záznamu dotazů se lze zaměřit na konkrétní dotazy a optimalizovat ty, které nejvíce vytěžují MySQL.

4.2. Implementace Pressflow 6

Implementace Pressflow 6 je součástí návrhu optimalizace výkonu demonstrativní stávající instalace CMS Drupal 6 pro portál iZUN.eu. Podle *tabulky č. 2: Srovnání Pressflow a Drupalu 6, 7* byla zvolena distribuce Pressflow 6. Splňuje požadavky kompatibility se stávající verzí CMS Drupal 6 a umožňuje využití pokročilých vlastností z běžné verze CMS Drupal 7.

4.2.1. Replikace databáze s Pressflow 6

Drupal 6 neumožňuje využít replikaci databáze MySQL. Pro replikaci databáze MySQL je nutné rozlišit přístupy k *master-serveru*, který umožňuje jak čtení, tak i zápis a k *slave-serveru* jen pro čtení. Pro přístup k databázi je využívána funkce *db_query()*, která nerozlišuje mezi *INSERT* pro vkládání dat a *SELECT* pro čtení dat. Například Drupal 7 v databázové vrstvě jádra systému rozlišuje funkce pro přístup k databázi *db_select()* a pro vkládání dat do databáze *db_insert()* (Drupal.org, 2014).

Pressflow 6 umožňuje přímo nastavit rozdílné přístupy k databázi v konfiguračním souboru (*/sites/default/settings.php*) pro master-server a slave-server:

```
$db_url = 'mysqli://uzivatel:heslo@master-host/databaze';
$db_slave_url = 'mysqli://uzivatel:heslo@slave-host/databaze';
```

V případě více slave-serverů jsou jejich přístupové údaje vloženy do pole:

```
$db_url = 'mysqli://uzivatel:heslo@master-host/databaze';
$db_slave_url = array();
$db_slave_url[] = 'mysqli://uzivatel:heslo@slave-host-1/databaze';
$db_slave_url[] = 'mysqli://uzivatel:heslo@slave-host-2/databaze';
```

Také v případě více master-serverů jsou přístupové údaje vloženy do pole a odlišeny *default* a *another*.

```
$db_url = array();
$db_url['default'] = 'mysqli://uzivatel:heslo@master-host/databaze';
$db_url['another'] = 'mysqli://uzivatel:heslo@another-host/databaze';

$db_slave_url = array();
$db_slave_url['default'] = array();
$db_slave_url['default'][] 'mysqli://uzivatel:heslo@slave-host-1/databaze';
$db_slave_url['default'][] 'mysqli://uzivatel:heslo@slave-host-2/databaze';
```

4.2.2. Profilování Drupal 6 a Pressflow 6

V rámci analýzy Pressflow 6 bylo provedeno sledování doby běhu jednotlivých částí systému a následné vyhodnocení. K této analýze PHP skriptů byl použit nástroj

XHProf[45], původně vyvinutý pro potřeby vývojového oddělení Facebooku v roce 2009. Tento nástroj disponuje jednoduchým grafickým HTML rozhraním. Zaznamenává dobu trvání, čas vykonání dané funkce a i funkcí z ní volaných, využití paměti, procesorového času a počet volání jednotlivých funkcí. (XHProf, 2015) Umožňuje také přímo srovnání výsledků dvou měření. Výsledky měření lze poté i zobrazit graficky s využitím vizualizační aplikace *dots* programu Graphviz[46].

Instalace XHProf a nastavení měření

XHProf lze stáhnout, nebo zkompilovat z repozitáře Pecl (XHProf, 2015) a následně nainstalovat jako rozšíření. Po úspěšné instalaci je nutné rozšíření zapnout a nastavit umístění adresáře pro uložení souborů s výsledky měření, uložením následujícího kódu (ř. č. 1 - 3) do konfiguračního souboru PHP *php.ini*:

```
1. [xhprof]
2. extension=xhprof.so
3. xhprof.output_dir=/tmp/xhprof
```

Začátek měření je zapnut funkcí *xhprof_enable(),* respektive pro měření se záznamem procesorového času funkcí *xhprof_enable(XHPROF_FLAGS_CPU + XHPROF_FLAGS_MEMORY).* Konec měření je proveden funkcí *xhprof_disable()* vracející pole volaných funkcí s naměřenými údaji. Toto pole je možné nechat zpracovat integrovaným grafickým rozhraním nástroje XHProf.

Níže je uveden zkrácený výpis souboru *index.php* s přidaným kódem pro začátek (řádek č. 2) a konec měření (ř. č. 5) a následný blok kódu (ř. č. 6 -12) pro zpracování výsledků a přidání odkazu pro zobrazení výsledků.

```
1. <?php
2. xhprof_enable(XHPROF_FLAGS_CPU + XHPROF_FLAGS_MEMORY)
3. …
4. drupal_page_footer();
5. $xhprofData = xhprof_disable();
6. $xhprofLibFolder = "/usr/local/apache2/htdocs/xhprof_lib";
7. include_once $xhprofLibFolder . "/utils/xhprof_lib.php";
8. include_once $xhprofLibFolder . "/utils/xhprof_runs.php";
9. $xhprofRuns = new XHProfRuns_Default();
10.$xhprofRunId = $xhprofRuns->save_run($xhprofData, "xhprof_foo");
11.echo "<a
12.href='http://localhost/xhprof_html/index.php?run=$xhprofRunId&source=xhprof_foo'>XHProf
HTML</a>";
```

[45] XHProf je hierarchický profiler PHP realizovaný v jazyce C jako rozšíření PHP pro operační systémy Linux, FreeBSD a Mac OS X. Dostupný je pod open.source licencí Apache 2.0. (2015)
[46] Graphviz - Graph Visualization Software (http://www.graphviz.org/, 20. 1. 2015)

Porovnání výsledků měření (tabulka č. 3: Profilování Drupal 6 a Pressflow 6) bylo provedeno na tzv. čisté instalaci CMS Drupal 6 a Pressflow 6 se zobrazením úvodní stránky (*node/1*) pro anonymního uživatele. Bylo provedeno 5 měření po sobě a výsledné hodnoty pro celkovou dobu trvání byly zprůměrovány. Grafické znázornění celého průběhu, návaznost volaných funkcí a ostatní měřené veličiny byly vloženy do *přílohy č. 5 Profilace Drupal 6* pro první měření Drupalu 6, respektive do *přílohy č. 6 Profilace Pressflow 6* pro první měření Pressflow 6.

Tabulka č. 3: Profilování Drupal 6 a Pressflow 6

	Počet volání funkcí	**Celková doba trvání (µs)***
Drupal 6	4030	52,748
Pressflow 6	4216	53,851

Zdroj: Autor **průměr měření*

Prvním výrazným rozdílem je vyšší počet volání funkcí Pressflow 6, při zachování podobného celkového času. Rozdíl je způsoben přepracováním jádra systému. Odlišný čas i u jednotlivých měření byl dán různým vytížením serveru (byť byl kladen důraz na co největší izolaci měřeného serveru), z tohoto důvodu byly naměřené hodnoty zprůměrovány.

4.2.3. Bezpečnost Pressflow

Pressflow obsahuje velmi rozsáhlý zásah do jádra systému CMS Drupal. Byť zachovává vnější kompatibilitu, je zcela kompatibilní s databází a je i API kompatibilní s jinými přidanými moduly. Přesto jakýkoliv zásah do jádra systému není z bezpečnostního hlediska doporučován (Trevor, 2010), ať už z pohledu stability systému, nebo např. možnosti zneužití případné chyby v kódu k neoprávněnému zásahu do aplikace.

4.3. Způsoby testování

Zátěžové testy simulující chování uživatele se skládají obvykle ze čtyř částí. První částí je tvorba scénáře, který vykonává virtuální uživatel. Tento vygenerovaný uživatel je naprogramovaný proces, který prochází daný scénář. Provádění jednotlivých scénářů množstvím souběžných procesů je pro jednotlivé virtuální uživatele posunuto v čase. Druhým krokem je nastavení testu, definování počtu a typu uživatelů, nebo zavedení

náhodné složky pro vyčkání. Následuje provedení testu, tedy spuštění hlavní testovací aplikace, případných jiných pomocných programů. Závěrečnou částí je srozumitelné zobrazení výstupů (tabulky, grafy) a jejich interpretace.

4.3.1. Rozdělení zátěžových testů

Zátěžové testy lze rozdělit do následujících kategorií pro testy v úplném rozsahu.

- Výkonnostní test (Performance Test)
- Test hraniční zátěže (Load/Stress Test)
- Test odolnosti (Soak Test)
- Test selhání (Failover Test)
- Test části infrastruktury (Target infrastructure Test)
- Test citlivosti sítě (Network Sensitivity Test)
- Test objemu dat (Volume Test)

Pro testy tzv. jediné komponenty, v tomto případě komponenty databáze MySQL, je důležité určit vhodnou metriku.

Výkonnostní testy zaměřené na databáze měří výkon následujícími metrikami. U transakčních databází, respektive transakčních úložných enginů se měří počet transakcí za jednotku času. Jedná se o tzv. měření *propustnosti*. Využívá se u víceuživatelských aplikací. Další mírou výkonu je doba odezvy, tedy *latence*; udává čas, po který byla daná úloha požadována. Krajní hodnota, maximální latence nenese dobrou vypovídající hodnotu. Protože s narůstající dobou provádění testu narůstá i pravděpodobnost prodloužení této hodnoty, není její měření ani opakovatelné (Schneiderx, 2009). Z tohoto důvodu je vhodnější využít pro měření doby odezvy percentil. Například naměření 95. percentilu latence rovného 30ms vypovídá, že s 95% pravděpodobností bude daná úloha ukončena do 30ms. *Souběžnost* na úrovni databáze udává počet simultánně běžících dotazů, což ale není počet připojení k databázi. Možné měření je např. kolik požadavků za jednu sekundu vytvoří uživatelé v provozní špičce serveru, tedy v době největší zátěže. B. Schwartz v MySQL profesionálně uvádí příklad, kdy se na webu nachází 50 000 uživatelů, ale server MySQL je zatěžován pouze 10 – 15 souběžně běžícími dotazy (Schwartz, a další, 2009). Posledním ukazatelem je *škálovatelnost*, která je nezbytná pro systémy zachovávající daný výkon i v případě měnící se pracovní zátěže.

4.4. Použité způsoby testování

4.4.1. AuthcacheFooter – statistika průběhu generování stránky

AuthcacheFooter je přídavný modul CMS Drupal, vytvářející JSON kód, který je připojen na konci stránky a obsahuje informace o času renderování stránky, času a počtu provedených dotazů v databázi, čas renderování cache, stáří cache aj. Zobrazení uvedených dat v JavaScriptovém okně je možné povolit (*Nastavení/povolení debug mode pro všechny role*). Zobrazení je vhodné pro bezprostřední porovnávání a kontrolu změn při jednotlivých úpravách.

4.4.2. Apache Benchmark

Apache Benchmark je program spouštěný z příkazové řádky. Provádí načtení stránky, značně zjednodušeně, bez grafiky, JS a dalších prvků. Program dokáže nasimulovat více souběžných připojení, a celé měření opakovat.

4.4.3. Apache JMeter

Apache JMeter je open-source program vytvořený v jazyce Java a umožňuje zátěžové testování, měření výkonnosti klient/server programů a webových aplikací. Program nefunguje jako webový prohlížeč, ale pracuje přímo na protokolové vrstvě. (JMeter Apache, 2015) Z tohoto důvodu nejsou spouštěny Javascripty. V tomto programu lze sestavit konkrétní test, který simuluje reálnou zátěž běžného provozu, tzn. určit počet klientů (anonymních/přihlášených), zasílané požadavky, nastavení intervalů aj. Test obsahuje virtuální uživatele, kteří provádějí předem připravené scénáře. Jednotlivé požadavky jsou nahrávány komponentou programu fungující podobně jako proxy server. Prostřednictvím této komponenty jsou nahrávány požadavky. Testy lze spouštět přímo v grafickém prostředí, v tomto případě může docházet k omezení výkonu spotřebovaném vykreslováním grafů, resp. tabulek. Pro získání relevantnějších dat bez zkreslení je vhodné spouštět program v konzoli.

4.5. Ověření navrhované optimalizace – zátěžové testy

4.5.1. Test zátěže serveru - Apache Benchmark

Pro test zátěže byl použit program Apache Benchmark. Tento program vytvoří daný počet souběžných připojení a celý postup *n* násobně opakuje. Výstupem je statistika úspěšnosti načtení stránky.

Vzorový příkaz: `ab -kc 10 -n 100 192.168.1.2/`

Program Apache Benchmark má přepínač *–c* pro počet souběžných připojení a přepínač *–n* pro počet opakování. (Apache http Server Version 2.2, 2015) Test byl proveden pro anonymní uživatele v deseti konkurenčních spojeních při http požadavku hlavní stránky.

Tabulka č. 4: Načtení hlavní stránky anonymním uživatelem, 10 konkurenčních připojení

	Propustnost	**Percentil 90 (ms)**
Bez paměti cache	18,39	852
Paměť cache normální	150,65	96
Paměť cache Boost	18,30	777

Zdroj: Autor

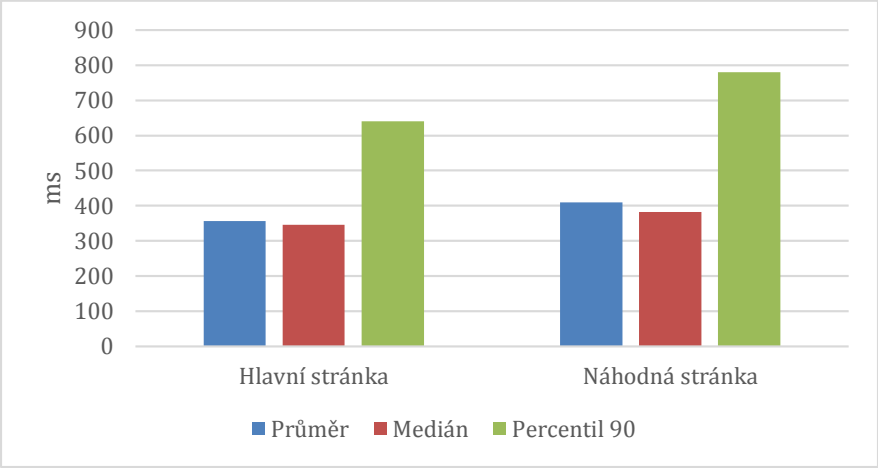
Uvedený způsob měření programem Apache Benchmark je značně zjednodušený. Přesnější výsledky měření lze dosáhnout programem Apache JMeter.

4.5.2. Test zátěže serveru – Apache JMeter

Apache JMeter umožňuje sestavit pokročilý test jak pro anonymního uživatele, tak i pro přihlášeného uživatele. Dále uvedené testy byly provedeny pro CMS Drupal 6 s popsanými parametry testů.

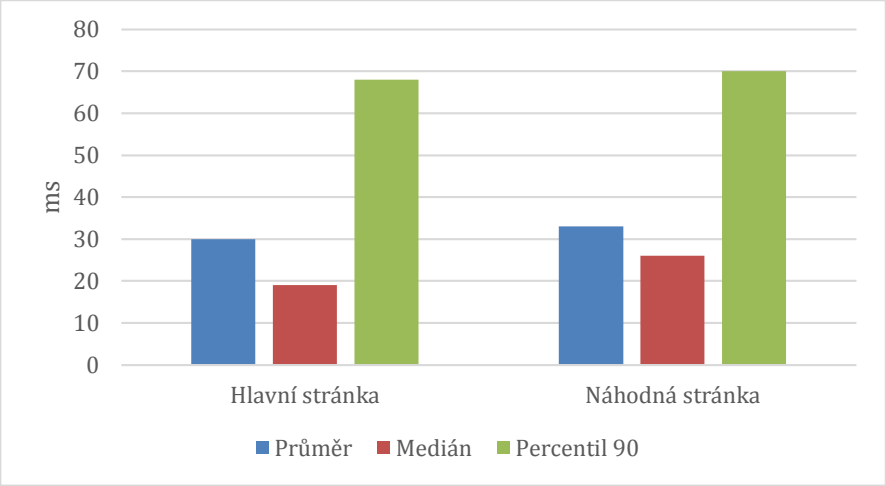
Anonymní uživatelé

Test pro nepřihlášené uživatele byl proveden vždy pro náhodné zobrazení jedné z 50 stránek a hlavní stránky, obsahující text a různý počet komentářů.



Obrázek č. 18: Čas načtení hlavní, náhodné str. anonymním uživatelem bez paměti cache, při 10 konkurenčních připojeních

Výsledky při použití normální paměti cache, uložené v databázi:



Obrázek č. 19: Čas načtení hlavní, náhodné str. anonymním uživatelem s normální pamětí cache, při 10 konkurenčních připojeních

Podle naměřených údajů při 10 konkurenčních připojeních najednou (simulující přístup 10 uživatelů) byl podle *obrázku č. 18: Čas načtení hlavní, náhodné str. anonymním uživatelem bez paměti cache, při 10 konkurenčních připojeních* 90. percentil roven 640ms pro hlavní stránku, resp. 780ms pro náhodnou stránku, propustnost[47] byla naměřena přibližně rovna 10. Všechny měřené údaje obsahuje následující tabulka č. 5.

[47] Výpočet podle vzorce = (počet požadavků) / (celkový čas).

Tabulka č. 5: Načtení hlavní, náhodné str. anonymním uživatelem, bez paměti cache, při 10 konkurenčních připojeních

	Vzorky	Průměr	Medián	Percentil 90	Min	Max	Propustnost	KB/sec
Hlavní stránka	395	356	345	640	66	834	10,12457	70,35578
Náhodná stránka	388	410	382	780	61	1170	9,984303	90,53336
Celkem	783	383	370	706	61	1170	20,06972	160,5341

Zdroj: Autor

Při použití integrované paměti cache na úrovni normální, *obrázek č. 19: Čas načtení hlavní, náhodné str. anonymním uživatelem s normální pamětí cache, při 10 konkurenčních připojeních*, bylo dosaženo 90. percentilu pro hlavní stránku 68ms, resp. 70ms pro náhodnou stránku. Při použití paměti cache v režimu normální byla v uvedeném měření získána úspora 637ms. Propustnost byla zvýšena více než jedenáctkrát. Naměřené hodnoty jsou uvedeny v následující *tabulka č. 6: Načtení hlavní, náhodné str. anonymním uživatelem, normální cache, při 10 konkurenčních připojeních.*

Tabulka č. 6: Načtení hlavní, náhodné str. anonymním uživatelem, normální cache, při 10 konkurenčních připojeních

	Vzorky	Průměr	Medián	Percentil 90	Min	Max	Propustnost	KB/sec
Hlavní stránka	4406	30	19	68	4	242	112,9773	777,5403
Náhodná stránka	4404	33	26	70	4	365	112,9347	1018,751
Celkem	8810	31	23	69	4	365	225,8569	1795,845

Zdroj: Autor

Při použití modulu Boost, který vytváří paměť cache statickými soubory, uloženými v souborovém systému serveru, byly naměřeny následující hodnoty v *tabulce č. 7: Načtení hlavní, náhodné str. anonymním uživatelem, cache modul Boost, 10 konkurenčních připojeních.*

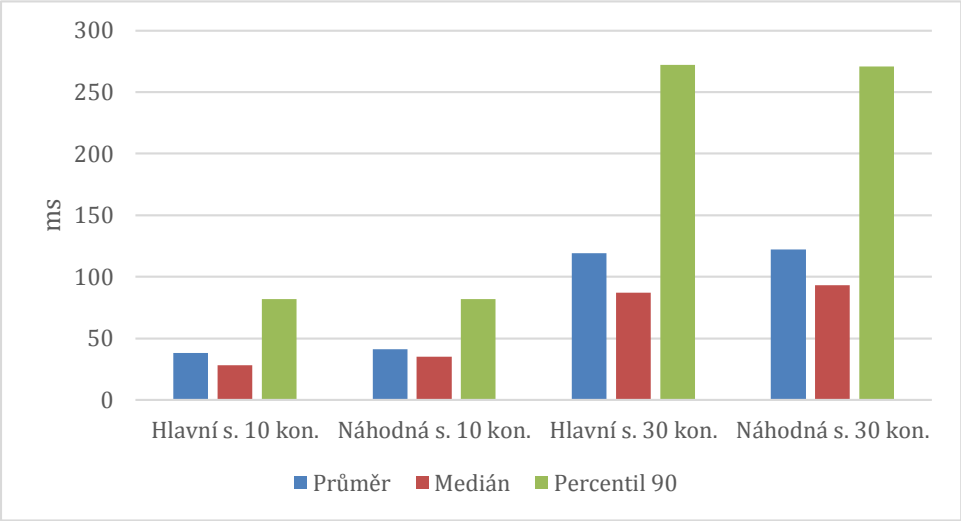
Tabulka č. 7: Načtení hlavní, náhodné str. anonymním uživatelem, cache modul Boost, 10 konkurenčních připojeních

	Vzorky	Průměr	Medián	Percentil 90	Min	Max	Propustnost	KB/sec
Hlavní stránka	3517	38	28	82	7	324	90,19568	620,7526
Náhodná stránka	3510	41	35	82	6	233	90,01847	811,6147
Celkem	7027	40	32	82	6	324	180,161	1431,943

Zdroj: Autor

Uvedené výstupy měření se výrazně neliší od použití integrované paměti cache v předešlém příkladu, nicméně jsou mírně vyšší (90 percentil roven 82ms je o celých 13ms vyšší). Významným přínosem je snížení zátěže kladené na databázi MySQL, na úkor

souborového systému serveru. Zde je možnost případné optimalizace (využití efektivních souborových systémů apod.). Užití statické paměti cache modulu Boost vede k úspoře systémových prostředků databázového serveru, které mohou být využity pro přihlášené uživatele.



Obrázek č. 20: Porovnání doby načtení náhodné str. anonym. uživatelem s pamětí cache modulu Boost, pro 10 a 30 konkurenčních připojení.

V průběhu testování nebyl zaznamenán žádný dotaz, který by překročil dobu trvání delší než 1 sekundu a byl by zapsán v logu-pomalých dotazů.

5. Zhodnocení výsledků a doporučení

Vývojový cyklus zahrnuje právě podporovanou verzi Drupalu 7 v režimu dlouhodobé podpory. Po jeho ukončení bude následovat jen podpora kritických bezpečnostních aktualizací. Dokončovaná verze 8 bude zahrnovat již zcela objektový přístup. Vývojový cyklus přináší každých 6 měsíců uvedení nové verze a pravidelné bezpečnostní aktualizace. V souvislosti s uváděním nových verzí autor zdůrazňuje omezení ve zpoždění dostupnosti aktualizace nových modulů.

V práci byly zhodnoceny možnosti optimalizace databáze a zdůrazněny možnosti vhodné pro použití v prostředí středně velké webové aplikace. Autor rozčlenil způsoby optimalizace podle způsobu správy webového serveru. V práci jsou rozčleněny postupy optimalizace v případě hostingu se značně omezenými právy správy, kdy je podle provedené analýzy doporučeno použití dostupného modulu (*DB Maintenance*). V případech plné správy serveru byly zhodnoceny postupy optimalizace v operačním systému LINUX - serveru MySQL s příklady použití programu *myisamchk*.

V rámci analýzy úložných enginů byla zkoumána možnost využití jiných typů než je primárně doporučováno pro CMS Drupal 6 v databázi MySQL. Zejména byla zkoumána možnost změny úložného enginu MyISAM pro ukládání paměti cache systému Drupal. Možnost využití velice rychlého úložného typu MEMORY se neprokázalo jako možné z důvodu specifických požadavků vzhledem ke struktuře dat a jejich objemu, systém Drupal využívá např. datové typy *longblob* a *text,* které analyzovaný úložný engin neumožňuje využít. Byla vyloučena i možnost změny datových typů na straně samotné aplikace.

Byly analyzovány možnosti využití replikace databáze a zhodnoceny konkrétní typy replikace asynchronní a semisynchronní. Podle způsobu přístupu k tabulkám nebo jejím částem v průběhu replikace bylo doporučeno použití typu asynchronní replikace v režimu MIXED. V případě použití transakcí, zvláště v případě vyšších verzí od CMS Drupal 7, autor došel k závěru využít semisynchronní replikace, která vede k vyšší spolehlivosti a konzistentnosti dat. Byla také analyzována možnost využití MySQL Clusteru, která zvyšuje spolehlivost až na teoreticky možných 99,999 % (přesněji přepočteno na výpadek serveru maximálně celkem 5 min/ročně). Toto řešení však obnáší vyšší hardwarové nároky. Na druhou stranu kromě vyšší spolehlivosti umožňuje

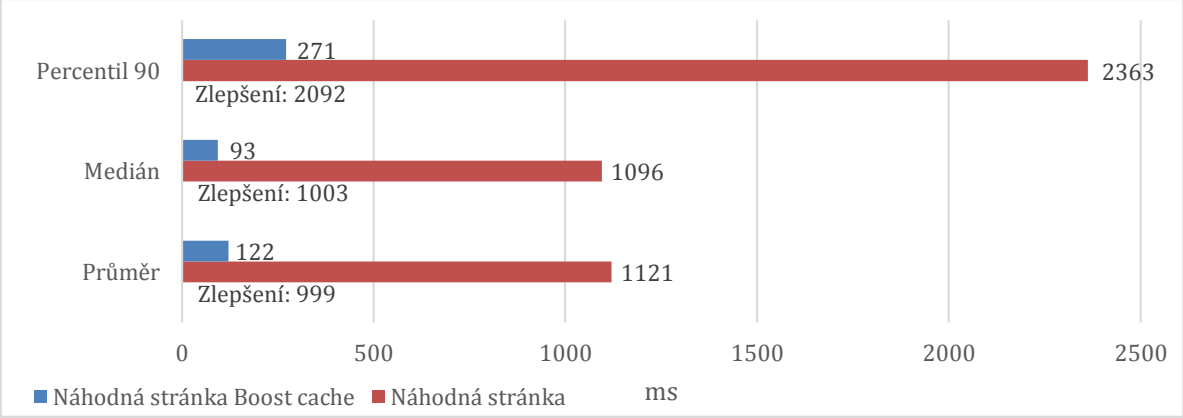
dynamicky za chodu měnit počet uzlů v clusteru, a tím reagovat na možný nárůst či pokles očekávaného výkonu. Přes zjištěné výhody, které jsou ovšem velice specificky úzce zaměřeny jen na některé případy, pro které je využití MySQL Clusteru výhodné, není toto řešení vhodné pro CMS Drupal.

Širší uplatnění pro zvýšení výkonu databáze CMS Drupal přináší využití replikace ve spojení s load-balancerem. Problematika rozložení zátěže společně s přímým připojením k replikovaným serverům byla popsána se zaměřením na způsob rozdělení a směrování dotazů pro zápis a čtení. Analýza rozložení zátěže vedla k položení otázky, zda se přiklonit pouze ke škálovatelnosti, anebo zvážit náročnější přístup, který vede k efektivnímu využití replikace a škálovatelnosti.

Řešení, jak plně využít přínos replikace v optimalizaci pro vysoký výkon databáze MySQL v kombinaci s CMS Drupal 6, je prostřednictvím Pressflow 6, které dokáže efektivně pracovat s replikačními servery.

Srovnání naměřených výsledků profilování CMS Drupal 6 a Pressflow 6 prokázalo značné přepracování jádra systému Pressflow 6 a jeho odlišnost. Naměřené hodnoty v základní instalaci obou sledovaných systémů vykazovaly obdobnou celkovou dobu trvání cca. 53µs.

Aplikace modulu Boost využívající paměť cache uloženou místo v databázi v souborovém systému vedla při zátěžovém testu jen k mírnému nárůstu zpoždění oproti výchozí paměti cache systému Drupal. Při 10 konkurenčních připojeních byl změřen 90. percentil 82ms.



Obrázek č. 21: Porovnání doby načtení náhodné str. anonym. uživatelem bez paměti cache a s pamětí cache modul Boost, 30 konkurenčních připojení.

Ucelený pohled na rozdíl při použití paměti cache a modulu Boost při vyšším počtu 30 konkurenčních připojení je uveden na *obrázku č. 21: Porovnání doby načtení náhodné str. anonym. uživatelem bez paměti cache a s pamětí cache modul Boost, 30 konkurenčních připojení.* V porovnání s 10 konkurenčními připojeními byl nárůst doby načtení náhodné stránky zvýšen pouze o 189ms (90. percentil).

6. Závěr

Redakční systém CMS Drupal je profesionální nástroj, který se rozvíjí za podpory silné komunity. Pro aktuálně podporovanou verzi Drupalu 7 je dostupný režim dlouhodobé podpory. Dokončovaná verze 8 bude zahrnovat již zcela objektový přístup. Vývojový cyklus přináší každých 6 měsíců uvedení nové verze a pravidelné bezpečnostní aktualizace.

V práci byly podrobně popsány a zhodnoceny metody optimalizace databáze MySQL v souvislosti s redakčním systémem Drupal 6. Byly analyzovány možnosti využití doplňkových modulů k optimalizaci využití databáze jak pro přihlášené uživatele, tak i pro anonymní uživatele. Dále také byly zhodnoceny přímo moduly optimalizující a provádějící údržbu databáze MySQL.

Pro anonymní uživatele byly provedeny výkonnostní testy při použití různých přístupů k paměti cache systému Drupal 6. Bylo dosaženo obdobných výsledků při použití standardní cache a speciální souborové cache modulu Boost.

Analýzou systému Drupal a struktury databáze byly zamítnuty změny typu uložiště databáze MySQL, a to i v případě jednotlivých částí databáze. Není možné využít pro zvýšení výkonu databáze jiné typy uložišť, nebyla prokázána ani vhodnost úložného typu MySQL Cluster. Autorem bylo ale doporučeno využití replikace databáze ve spojení se speciální distribucí Drupalu Pressflow 6. Ta je vhodná pro použití v aplikacích vyžadujících zvýšení výkonu. Je plně kompatibilní s CMS Drupal 6 a umožňuje podporu integrace s technologiemi na aplikační a prezentační vrstvě serveru vedoucí ke zvýšení výkonu.

Oba systémy Drupal 6 a Pressflow 6 byly porovnány využitím profilování a následně základním zátěžovým testem byl ověřen jejich výkon. Oblast optimalizace databáze je velmi úzká a prostor pro výraznější zvýšení výkonu poskytuje právě oblast prezentační a aplikační, které byly v práci také analyzovány. Autor dále upozornil na možné bezpečnostní riziko při použití distribuce Pressflow, která obsahuje zásah do kódu jádra systému.

7. Seznam použitých zdrojů

Drupal.org. [Online]. 25. 10. 2014. Dostupné z WWW <http://www.drupal.org>.

Apache HTTP Server Version 2.2. [Online]. 14. 1. 2015. Dostupné z WWW <http://httpd.apache.org/docs/2.2/programs/ab.html>.

AgriDrupal. *Agricultural information management using the Drupal CMS*. [Online]. 5. 12. 2014. Dostupné z WWW <http://www.agridrupal.org/>.

AIMS. *Agricultural Information Management Standards*. [Online]. 12. 10. 2014. Dostupné z WWW <http://aims.fao.org/about>.

BUYTAERT, Dries. *Proposal to manage the Drupal core release cycle*. Drupal. [Online]. 20. 1. 2015. Dostupné z WWW <https://www.drupal.org/node/2135189>.

CARPER, M. Boost. *Drupal.org*. [Online]. 20. 1. 2015. Dostupné z WWW <https://www.drupal.org/project/boost>.

Comparison Pressflow, *Drupal*. [Online]. 28. 1 2015. Dostupné z WWW <https://pressflow.atlassian.net/wiki/display/PF/Comparison+ +Pressflow+versus+Drupal>.

DAVIES, A. a Fisk, H. MySQL Clustering. Indianapolis: MySQL Press, 2006. ISBN 0-672-32855-0.

Drupal not CMS. [Online]. 20. 11. 2014. Dostupné z WWW <http://www.palantir.net/blog/drupal-not-cms>.

ELLISON, Jonah. Authenticated User Page Caching. Drupal.org. [Online]. 20. 12. 2014. Dostupné z WWW <https://www.drupal.org/project/authcache>.

Explaining architectural tiers Drupal. [Online]. 4. 12. 2014. Dostupné z WWW <http://robknight.org.uk/blog/2011/02/explaining-architectural-tiers-drupal>.

GILMORE, J. W. *Velká kniha PHP a MySQL 5: kompendium znalostí pro začátečníky i profesionály*. Brno : Zonner Press, 2007. ISBN 978-80-86815-53-4.

HODGDON, J. *Programmer's Guide to Drupal*. O'Reilly Media, Inc.,, 2012. ISBN 978-1-449-34331-6.

CHAMBERLIN, D. D. a Boyce, R. F. *Sequel: A Structured English Query Language.* [Online]. 10. 2. 2015. Dostupné z WWW <http://www.almaden.ibm.com/cs/people/chamberlin/sequel-1974.pdf>.

January 2015 Web Server Survey. Netcraf. [Online]. 1. 2. 2015. Dostupné z WWW <http://news.netcraft.com/archives/2015/01/15/january-2015-web-server-survey.html>.

JMeter Apache. [Online]. 10. 1. 2015. Dostupné z WWW <http://jmeter.apache.org/>.

KOFLER, M. *Mistrovství v MySQL 5.* Brno: Computer Press, a.s., 2007. ISBN 978-80-251-1502-2.

MySQL 5.6 Reference Manual. *MySQL*. [Online]. 5. 1. 2015. Dostupné z WWW <http://dev.mysql.com/doc/refman/5.6/en/>.

NARAYAN, N., Newton, N. a Catchpole, N. *High Performance Drupal. Sebastopol*: O'Reilly, 2013. ISBN 978-1-449-39261-1.

O systému Drupal. *Drupal.cz*. [Online]. 3. 10. 2014. Dostupné z WWW <https://www.drupal.cz/o-systemu-drupal>.

PEKÁREK, V. *Zabezpečení a správa přístupů MySQL. Praha*: ČZU, 2012.

POLZER, J. *333 tipů a triků pro Drupal.* Brno: Computer Press, a.s., 2010. ISBN 978-80-251-2942-5.

POLZER, J. *Drupal 7 Podrobný průvodce tvorbou a správou webů*. Brno: Computer Press, a.s., 2011. ISBN 978-80-251-3445-0.

Pressflow documentation. [Online] 19. 12. 2014. Dostupné z WWW <https://pressflow.atlassian.net/wiki/display/PF/Pressflow+documentation/>.

PRESTON-WERNER, Tom. *Semantic Versioning 2.0.0.* [Online]. 10. 2. 2015. Dostupné z WWW <http://semver.org/>.

SCHNEIDERX, R. *MySQL Oficiální průvodce tvorbou, správou a laděním databází*. Praha: Grada Publishing, a.s., 2009. ISBN 80-247-1516-3.

SCHWARTZ, B., a další. *MySQL profesionálně - optimalizace pro vysoký výkon*. Brno: Zoner software, a.s., 2009. ISBN 978-80-7413-035-9.

System reguirements. [Online]. 5. 11. 2014. Dostupné z WWW <http://drupal.org/requirements>.

TREVOR, J. *Drupal 6 Performance Tips. Birmingham*: Packt Publishing, 2010. ISBN 978-1-847195-84-5.

VALENTA, M. [Online] 23. 9. 2014. Dostupné z WWW <http://service.felk.cvut.cz/courses/Y36DBA/slides-2009/dba_04_mysql.pdf>.

VANDYK, J. *Pro Drupal Development*. Apress, 2008. ISBN 978-1430209898.

Whitepaper, *A MySQL*. *Scaling Continuiusly-Available Web Services.* [Online]. 25. 1. 2015. Dostupné z WWW <http://www.mysql.com>.

XHProf. *XHProf: A Hierarchical Profiler for PHP*. [Online]. 25. 1. 2015. Dostupné z WWW <http://pecl.php.net/package/xhprof>.

8. Přílohy

8.1. Seznam obrázků a grafů

Obrázek č. 1: Základní přehled architektury CMS Drupal (VanDyk, 2008) 15

Obrázek č. 2: Architektura CMS Drupal (Explaining architectural tiers Drupal, 2014) 16

Obrázek č. 3: Přehled zpracování DRUPAL HTTP požadavku (Hodgdon, 2012) 19

Obrázek č. 4: Souborová struktura CMS Drupal 21

Obrázek č. 5: Požadavky podle typu MIME (měření 10. 1. 2015 http://www.webpagetest.org/) 22

Obrázek č. 6: Vývojový cyklus jádra CMS Drupal (Buytaert, 2015) 23

Obrázek č. 7: Tabulky v databázi pro uložení cache 28

Obrázek č. 8: Boost - rozdíl přihlášeného/nepřihlášeného uživatele (Carper, 2015) 30

Obrázek č. 9: Authcache - rozdíl pro přihlášeného/nepřihlášeného uživatele (Ellison, 2014) 31

Obrázek č. 10: Funkce Varnish reverzní proxy (Narayan, Newton, & Catchpole, 2013) .. 34

Obrázek č. 11: Schéma MySQL serveru (Valenta, 2014) 38

Obrázek č. 12: tabulka cache - struktura 42

Obrázek č. 13: tabulka cache - data 42

Obrázek č. 14: Replikace pro zvýšení výkonu, škálovatelnost (MySQL 5.6 Reference Manual, 2015) 45

Obrázek č. 15: Komplexní případ využití Load balancingu a replikace MySQL databáze 48

Obrázek č. 16: MySQL Cluster - vztah jednotlivých komponent (MySQL 5.6 Reference Manual, 2015) 50

Obrázek č. 17: Možnosti konfigurace Clusteru kombinací škálovatelnosti, vysoké dostupnosti a odolnosti (Whitepaper, 2015) 51

Obrázek č. 18: Čas načtení hlavní, náhodné str. anonymním uživatelem bez paměti cache, při 10 konkurenčních připojení 59

Obrázek č. 19: Čas načtení hlavní, náhodné str. anonymním uživatelem s normální pamětí cache, při 10 konkurenčních připojení 59

Obrázek č. 20: Porovnání doby načtení náhodné str. anonym. uživatelem s pamětí cache modulu Boost, pro 10 a 30 konkurenčních připojení. 61

Obrázek č. 21: Porovnání doby načtení náhodné str. anonym. uživatelem bez paměti cache a s pamětí cache modul Boost, 30 konkurenčních připojení. 63

8.2. Seznam tabulek

Tabulka č. 1: Moduly pro cache v CMS Drupal 6 - srovnání modulů 29

Tabulka č. 2: Srovnání Pressflow a Drupalu 6, 7 35

Tabulka č. 3: Profilování Drupal 6 a Pressflow 6 55

Tabulka č. 4: Načtení hlavní stránky anonymním uživatelem, 10 konkurenčních připojení 58

Tabulka č. 5: Načtení hlavní, náhodné str. anonymním uživatelem, bez paměti cache, při 10 konkurenčních připojení 60

Tabulka č. 6: Načtení hlavní, náhodné str. anonymním uživatelem, normální cache, při 10 konkurenčních připojení 60

Tabulka č. 7: Načtení hlavní, náhodné str. anonymním uživatelem, cache modul Boost, 10 konkurenčních připojení 60

8.3. Seznam příloh

Příloha č. 1: Načtení stránky izun.eu, vodopádový model načítání (měření 10. 1. 2015 http://www.webpagetest.org/)

Příloha č. 2: Připojení, vodopádový model připojení (měření 10. 1. 2015 http://www.webpagetest.org/)

Příloha č. 3: Složení typů požadavků pro jednotlivé typy souborů (měření 10. 1. 2015 http://www.webpagetest.org/)

Příloha č. 4: Podíl na celkové velikosti podle typů souborů (měření 10. 1. 2015 http://www.webpagetest.org/)

Příloha č. 5: Profilace Drupal 6

Příloha č. 6: Profilace Pressflow 6

Příloha č. 7: Plán testu č. 1 – Drupal 6, iZUN.eu

Příloha č. 1: Načtení stránky izun.eu, vodopádový model načítání (měření 10. 1. 2015 http://www.webpagetest.org/)

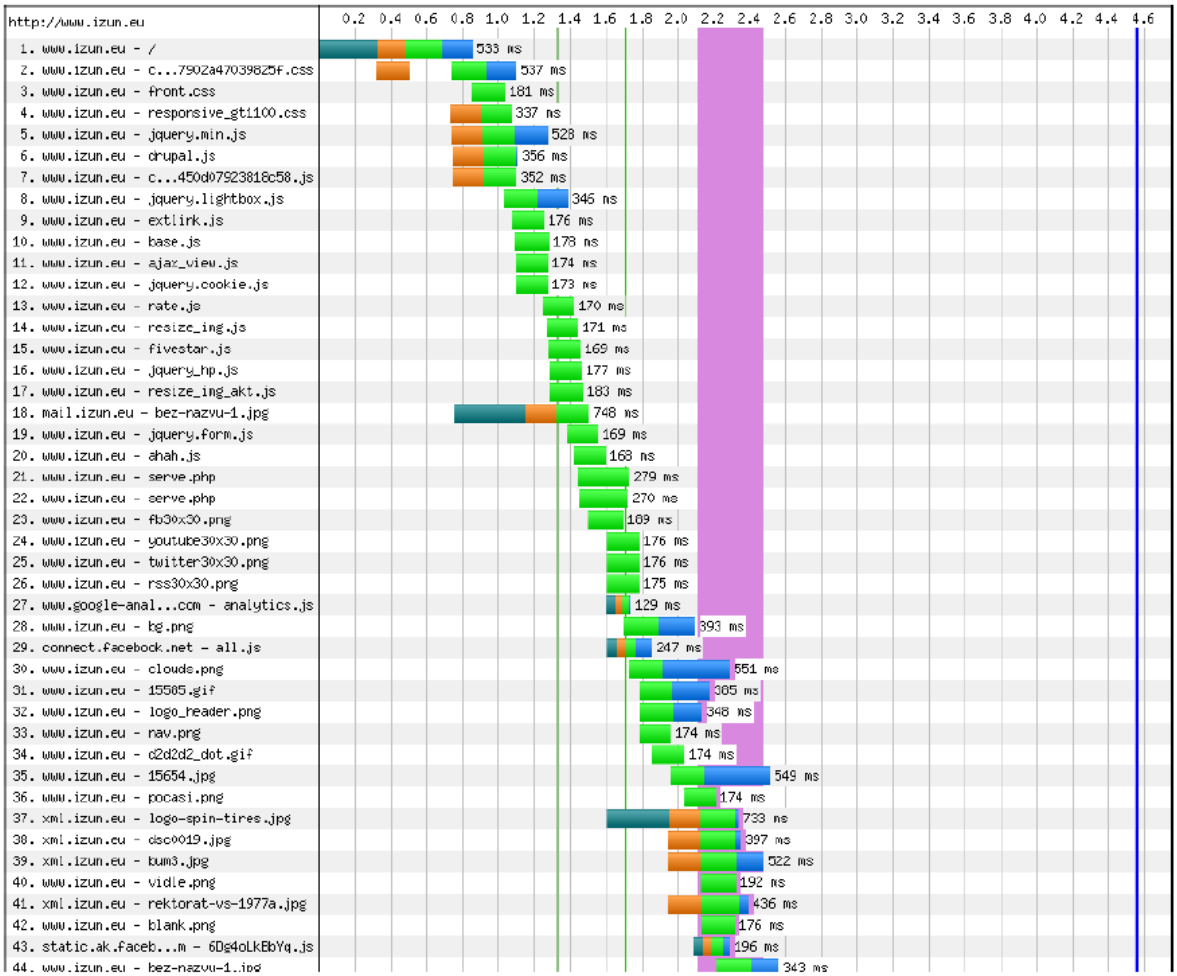
							Document Complete			Fully Loaded		
Load Time	First Byte	Start Render	Visually Complete	Speed Index	DOM Elements	Result (error code)	Time	Requests	Bytes In	Time	Requests	Bytes In
4.561s	0.685s	1.705s	6.600s	2263	1891	0	4.561s	131	516 KB	4.736s	131	883 KB

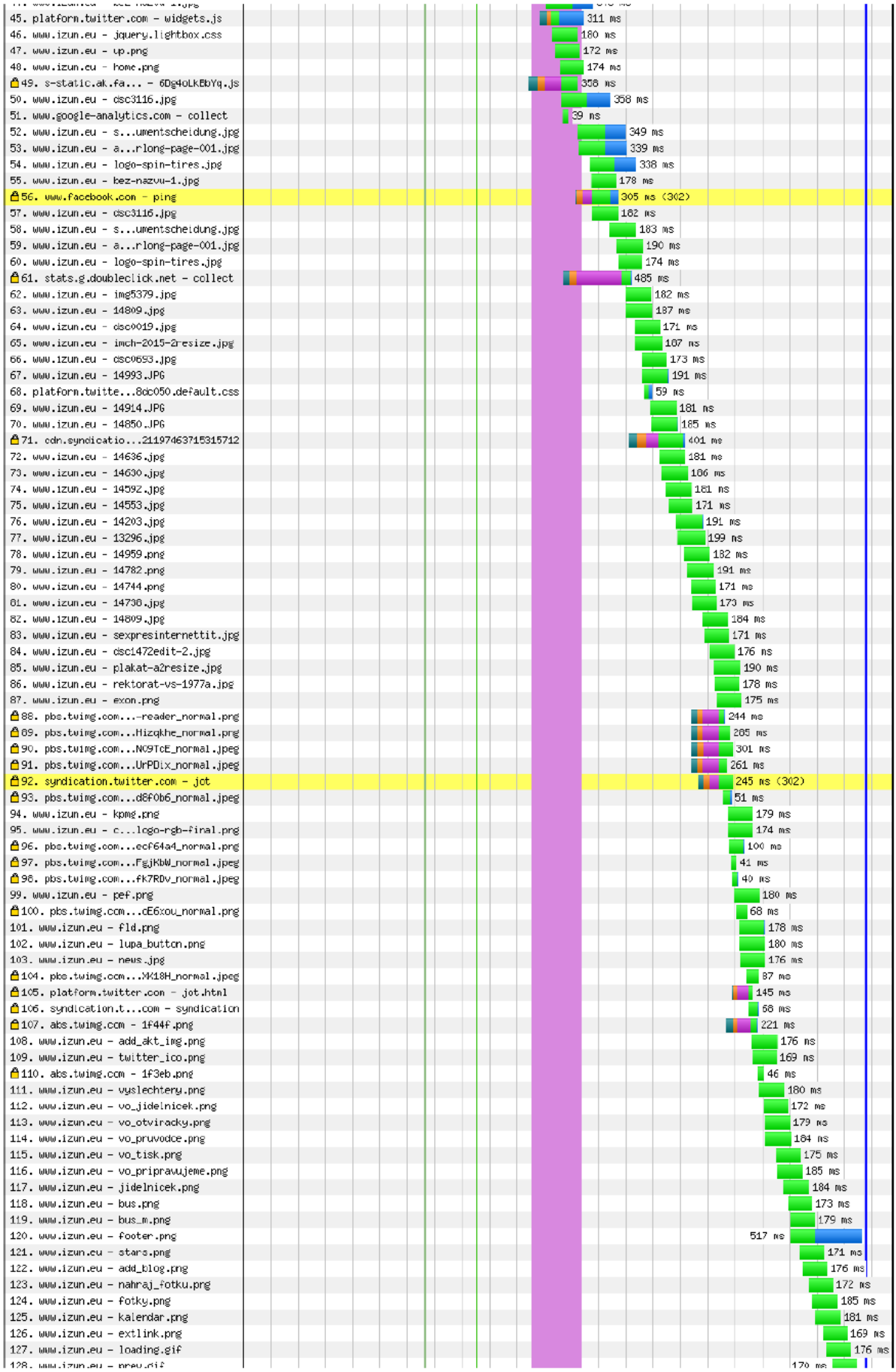
RUM First Paint	domContentLoaded	loadEvent
1.329s	2.111s - 2.470s (0.359s)	4.550s - 4.556s (0.006s)

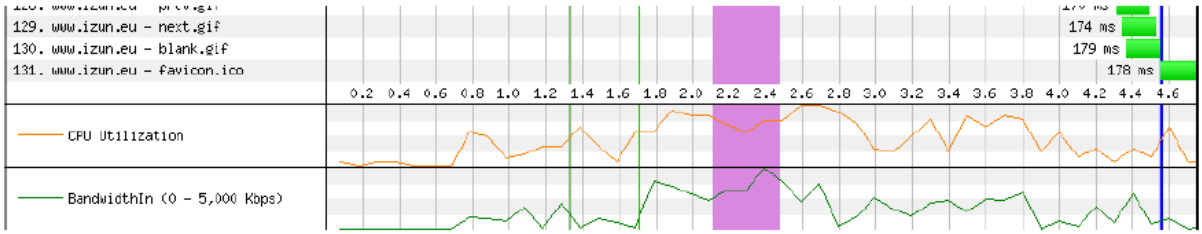
Waterfall View

DNS Lookup | Initial Connection | SSL Negotiation | Time to First Byte | Content Download | 3xx response | 4xx+ response

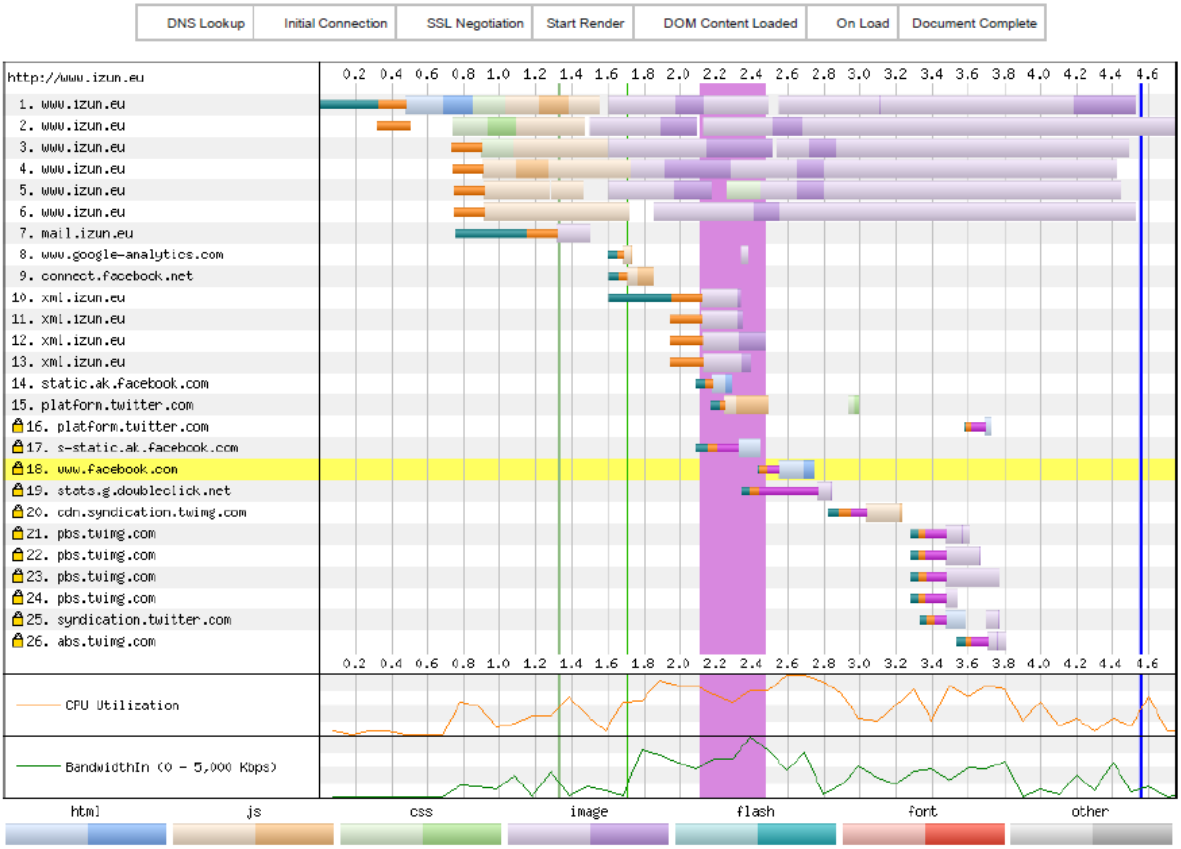
Start Render | msFirstPaint | DOM Content Loaded | On Load | Document Complete



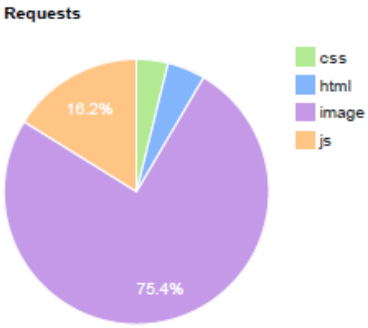

45. platform.twitter.com - widgets.js 311 ms
46. www.izun.eu - jquery.lightbox.css 180 ms
47. www.izun.eu - up.png 172 ms
48. www.izun.eu - home.png 174 ms
49. s-static.ak.fa... - 6Dg4oLkBbYq.js 358 ms
50. www.izun.eu - dsc3116.jpg 358 ms
51. www.google-analytics.com - collect 39 ms
52. www.izun.eu - s...umentscheidung.jpg 349 ms
53. www.izun.eu - a...rlong-page-001.jpg 339 ms
54. www.izun.eu - logo-spin-tires.jpg 338 ms
55. www.izun.eu - bez-nazvu-1.jpg 178 ms
56. www.facebook.com - ping 305 ms (302)
57. www.izun.eu - dsc3116.jpg 182 ms
58. www.izun.eu - s...umentscheidung.jpg 183 ms
59. www.izun.eu - a...rlong-page-001.jpg 190 ms
60. www.izun.eu - logo-spin-tires.jpg 174 ms
61. stats.g.doubleclick.net - collect 485 ms
62. www.izun.eu - img5379.jpg 182 ms
63. www.izun.eu - 14809.jpg 187 ms
64. www.izun.eu - dsc0019.jpg 171 ms
65. www.izun.eu - imch-2015-2resize.jpg 187 ms
66. www.izun.eu - dsc0653.jpg 173 ms
67. www.izun.eu - 14993.JPG 191 ms
68. platform.twitte...8dc050.default.css 59 ms
69. www.izun.eu - 14914.JPG 181 ms
70. www.izun.eu - 14850.JPG 185 ms
71. cdn.syndicatio...21197463715315712 401 ms
72. www.izun.eu - 14636.jpg 181 ms
73. www.izun.eu - 14630.jpg 186 ms
74. www.izun.eu - 14592.jpg 181 ms
75. www.izun.eu - 14553.jpg 171 ms
76. www.izun.eu - 14203.jpg 191 ms
77. www.izun.eu - 13296.jpg 199 ms
78. www.izun.eu - 14959.png 182 ms
79. www.izun.eu - 14782.png 191 ms
80. www.izun.eu - 14744.png 171 ms
81. www.izun.eu - 14738.jpg 173 ms
82. www.izun.eu - 14809.jpg 184 ms
83. www.izun.eu - sexpresinternettit.jpg 171 ms
84. www.izun.eu - dsc1472edit-2.jpg 176 ms
85. www.izun.eu - plakat-a2resize.jpg 190 ms
86. www.izun.eu - rektorat-vs-1977a.jpg 178 ms
87. www.izun.eu - exon.png 175 ms
88. pbs.twimg.com...-reader_normal.png 244 ms
89. pbs.twimg.com...Hizqkhe_normal.png 265 ms
90. pbs.twimg.com...NC9TcE_normal.jpeg 301 ms
91. pbs.twimg.com...UrPDix_normal.jpeg 261 ms
92. syndication.twitter.com - jot 245 ms (302)
93. pbs.twimg.com...d8f0b6_normal.jpeg 51 ms
94. www.izun.eu - kpmg.png 179 ms
95. www.izun.eu - c...logo-rgb-final.png 174 ms
96. pbs.twimg.com...ecf64a4_normal.png 100 ms
97. pbs.twimg.com...FgjKbW_normal.jpeg 41 ms
98. pbs.twimg.com...fk7RDv_normal.jpeg 40 ms
99. www.izun.eu - pef.png 180 ms
100. pbs.twimg.com...cE6xou_normal.png 68 ms
101. www.izun.eu - fld.png 178 ms
102. www.izun.eu - lupa_button.png 180 ms
103. www.izun.eu - news.jpg 176 ms
104. pbs.twimg.com...MK18H_normal.jpeg 87 ms
105. platform.twitter.com - jot.html 145 ms
106. syndication.t...com - syndication 58 ms
107. abs.twimg.com - 1f44f.png 221 ms
108. www.izun.eu - add_akt_img.png 176 ms
109. www.izun.eu - twitter_ico.png 169 ms
110. abs.twimg.com - 1f3eb.png 46 ms
111. www.izun.eu - vyslechtery.png 180 ms
112. www.izun.eu - vo_jidelnicek.png 172 ms
113. www.izun.eu - vo_otviracky.png 179 ms
114. www.izun.eu - vo_pruvodce.png 184 ms
115. www.izun.eu - vo_tisk.png 175 ms
116. www.izun.eu - vo_priprawujeme.png 185 ms
117. www.izun.eu - jidelnicek.png 184 ms
118. www.izun.eu - bus.png 173 ms
119. www.izun.eu - bus_m.png 179 ms
120. www.izun.eu - footer.png 517 ms
121. www.izun.eu - stars.png 171 ms
122. www.izun.eu - add_blog.png 176 ms
123. www.izun.eu - nahraj_fotku.png 172 ms
124. www.izun.eu - fotky.png 185 ms
125. www.izun.eu - kalendar.png 181 ms
126. www.izun.eu - extlink.png 169 ms
127. www.izun.eu - loading.gif 176 ms



Příloha č. 2: Připojení, vodopádový model připojení (měření 10. 1. 2015 http://www.webpagetest.org/)

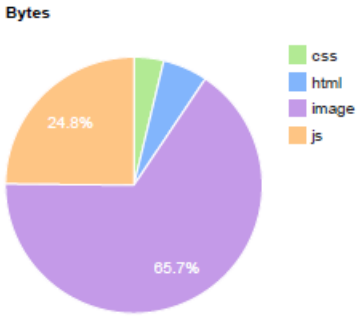


Příloha č. 3: Složení typů požadavků pro jednotlivé typy souborů (měření 10. 1. 2015 http://www.webpagetest.org/)



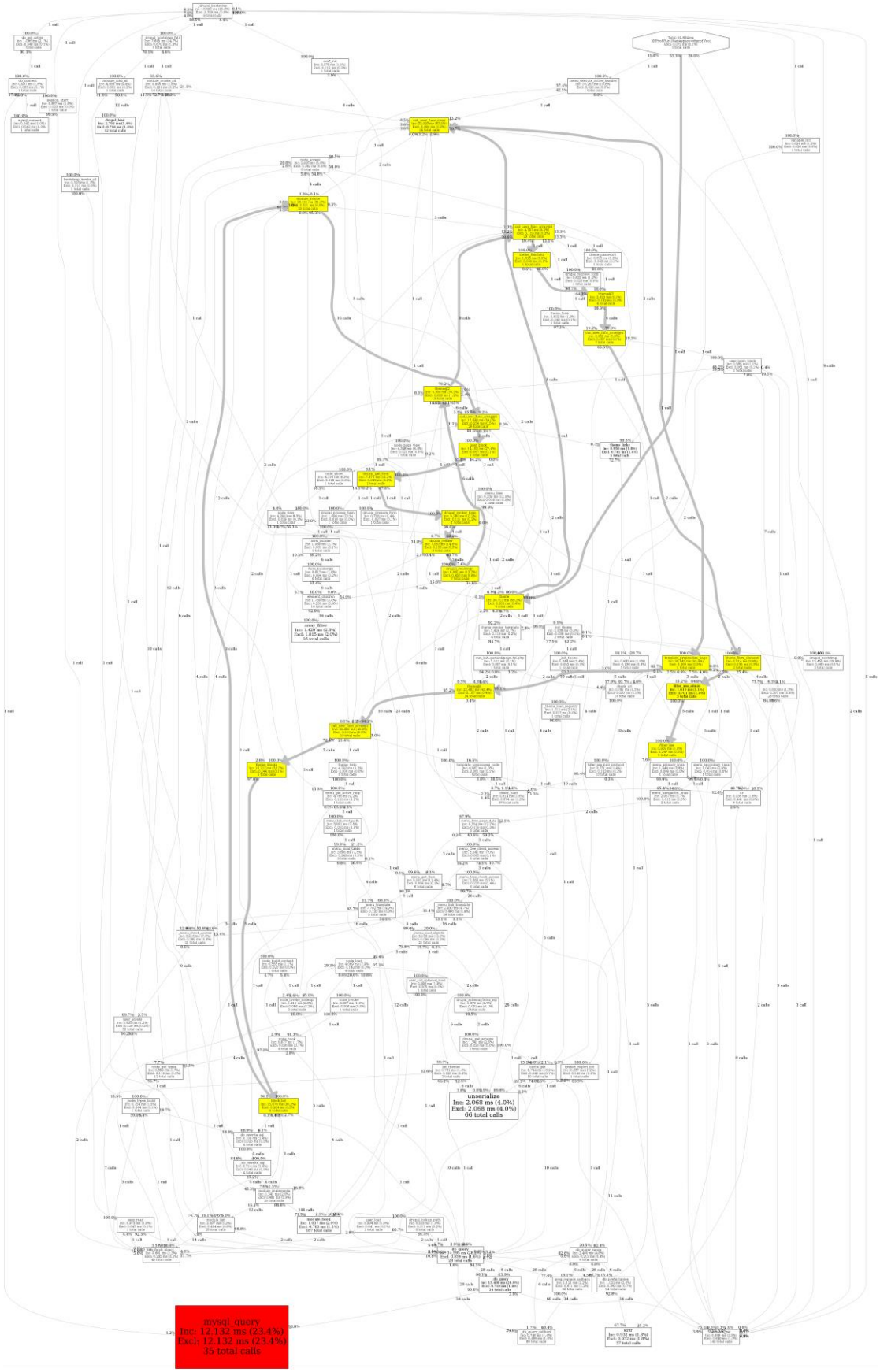
MIME Type	Requests
image	98
js	21
html	6
css	5
flash	0
font	0
other	0

Příloha č. 4: Podíl na celkové velikosti podle typů souborů (měření 10. 1. 2015 http://www.webpagetest.org/)



MIME Type	Bytes
image	567262
js	214389
html	48729
css	32384
flash	0
font	0
other	0

Příloha č. 5: Profilace Drupal 6



Příloha č. 6: Profilace Pressflow 6