PËRMIRËSIMI I PERFORMANCËS NË APLIKACIONE INTERNETI TË PASURA

Eljona Proko
Universiteti "Ismail Qemali" Vlorë. *E-mail: elzavalani@gmail.com*

Abstrakt
Gjatë këtyre tre dekadave të fundit World Wide Web ka pësuar një evolucion të vazhdueshëm. Për tu përgjigjur kërkesave gjithmonë e në rritje në fushën e aplikacioneve web janë propozuar vazhdimisht standarte dhe teknika me synimin për të zhvilluar aplikacione akoma më efektive dhe me performancë të lartë. Këto teknologji të reja kanë bërë të mundur lindjen e një gjenerate të re aplikacionesh web të quajtura aplikacione interneti të pasura (Rich Internet Applications). Aplikacione interneti të pasura janë të ndërtuara duke përdorur teknologji të suksesshme si Ajax, Flex, Flash dhe Silverlight. Qëllimi i këtij studimi është matja dhe analiza e performancës si një nga çështjet kryesore në një aplikacion. Studimi i performancës së aplikacioneve web është një proces kompleks i cili varet nga faktorë të ndryshëm. Metodologjia e përdorur në këtë studim është testimi i teknikes Query Cache e përdorur kryesisht në bazë të dhënash MySql. Në mënyrë që të ketë një pamje të qartë të efikasitetit të kësaj teknike, përveç shpjegimit të konfigurimit, ne do të bëjmë një krahasim me të dhënat reale, para dhe pas aktivizimit të kësaj teknike. Testimi është realizuar mbi një aplikacion të zhvilluar me anë të teknologjisë Ajax. Nga matjet e kryera rezulton se në seksionet me më shumë Query për tu ekzekutuar ka një ndryshim të ndjeshëm mes aktivizimit dhe mosaktivizimit të QueryCache. Duke u bazuar në eksperiencën e gjatë në projektimin dhe administrimin e aplikacioneve web dhe në rezultatet e matjeve të bëra nxjerrim si konkluzion që QueryCache është një teknike e dobishme dhe e kështillueshme për tu përdorur në të gjitha aplikacionet web ku si back-end përdoret MySQL. Kjo teknikë përmirëson dukshëm performancën e aplikacionit.

Fjalë Kyçe - Aplikacione interneti të pasura, Ajax, performancë, teknika Cache

1. Hyrje

Platformat web ofrojnë avantazhe të mëdha për zhvillimin e bizneseve, komunikimin, transmetimin e të dhënave në kohë reale, por ekzistojnë probleme në lidhje me sigurinë e të dhënave, vazhdueshmërin, performancën, dhe çështje të tjera. Performanca [1] është një nga çështjet kryesore në një aplikacion, e cila duhet studiuar me kujdes që në fillim të projektit dhe përgjatë zhvillimit të tij. Studimi i performancës së aplikacioneve web është një proces kompleks i cili varet nga faktor të ndryshëm dhe zgjidhjet e ofruara nga specialistë dhe nga kompani të ndryshme në vlerësimin e një produkti nuk janë të njëjta. Kjo gjë është e lidhur me platformën e përdorur, bazën e të dhënave, topologjinë e aplikacionit, etj. Edhe brenda të njëjtës platformë, ka një sërë teknikash të përdorura për vlerësimin e një platforme. Metodologjia e përdorur në këtë studim është testimi i teknikës Query Cache [2] e përdorur kryesisht në bazë të dhënash MySql [3]. Në mënyrë që të ketë një pamje të qartë të efikasitetit të kësaj teknike, përveç shpjegimit të konfigurimit të tij, ne do të bëjmë një krahasim me të dhënat reale, para dhe pas aktivizimit të kësaj teknike. Testimi është realizuar mbi një aplikacion të përdorur në sisteme bankare M.I.S. Në seksionin e dytë do të prezantojmë një vështrim të përgjithshëm mbi aplikacionet interneti të pasura. Në seksionin e tretë do të paraqesim metodologjinë eksperimentale, për

të vazhduar me seksionin e fundit ku japim rezultatet e testimit dhe konkluzionet

2. Aplikacione interneti te pasura

Në fillimet e tij World Wide Web ishte një platformë me anë të së cilës mund të kishe akses në përmbajtjet statike ose dinamike të koduara në HyperText Markup Language (HTML). Përdoruesit ishin të kufizuar në ndërveprime të navigonin nëpër linke dhe hedhjen e të dhënave në forma. Zgjidhjet moderne Web i ngjajnë aplikacioneve desktop [7], duke mundësuar ndërveprim të sofistikuar për përdoruesit, proçesim në anën e klientit, komunikim asinkron, dhe multimedia. Një rol të rëndësishëm në ditët e sotme luajnë aplikacionet internet të pasura (Rich Internet Applications, RIA). Termi RIA i referohet një familje heterogjene të zgjidhjeve, të karakterizuar nga një qëllim i përbashkët për të shtuar kapacitete të reja Web-it. Në RIA kombinohen lehtë arkitektura e shpërndarjes Web [4] me ndërfaqet interaktive të aplikimeve desktop dhe fuqinë llogaritëse. Ky kombinim përmirëson të gjitha elementët e një aplikimi Web (të dhënat, logjika e biznesit, komunikimi dhe prezantimi). Aplikacione interneti të pasura zhvillohen duke përdorur teknikat dhe teknologjitë Web 2.0, si Ajax [5], ose platforma Ajax si ASP.NET Ajax [6], ose platforma jo Ajax si Microsoft Silverlight, Adobe AiR, Adobe Flex, etj. Përdorimi i teknologjive RIA i krijon lehtësi përdoruesëve dhe ndërveprim me aplikacionet web. Por kjo vjen e gjitha me një kosto, që do të thotë përdorimi i RIA ngre çështjet që përfshijnë gjuhën dhe standardet arkitekturore që përdoruren për të zhvilluar këto aplikacione. Një çështje specifike kritike për RIA është gjetja e metodave dhe teknologjive të përshtatshme për të mbështetur të gjitha aktivitetet gjatë ciklit të jetës së software në mënyrë efektive, si dhe mbi të gjitha aktivitetet e mirëmbajtjes dhe testimit. Për çdo tip aplikacioni software dihet rëndësia e proçesit të mirëmbajtjes. Megjithatë, këto probleme janë edhe më të rëndësishme në kontekstin e RIAs, pasi këto aplikacione janë zhvilluar zakonisht në kohë të shkurtër nga programuesit që nuk përdorin praktika të njohura të Inxhinierisë Software, dhe shpesh përdorin strukturat dhe mjetet që, nga njëra anë thjeshtojnë zhvillimin e RIAs, nga ana tjetër prodhojnë kode komplekse që janë të vështira për tu kuptuar, duke cënuar cilësinë e produktit përfundimtar. Për më tepër, të dyja si natyra asinkrone dhe heterogjene të RIAs, të cilat janë të zhvilluara me anë të teknologjive të ndryshme dhe janë të bazuara në modelin e arkitekturës klient-server në të cilin komunikimi ndërmjet klientit dhe serverit mund të jetë asinkron, bëjnë të vështirë për të kuptuar RIA [10] dhe rrjedhimisht të vështirë për të mirëmbajtur dhe testuar [8].

3. Metodologjia Eksperimantale

Sistemi i Menaxhimit të Informacionit (Management Information System, MIS) është një software menaxhimi me base web-i, i përdorur në sistemin bankar, të specializuara në sektorin NPL (non performing loan). Ky program nuk përdoret vetëm për ruajtjen e të dhënave, por gjithashtu merr rezultate në kohë reale, dhe jep parashikime bazuar në ngjarjet që kanë ndodhur më parë. Programi komunikon në përputhshmëri me të gjitha programet e tjera të ndryshimit të të dhënave në të gjitha formatet e njohura të bazave të të dhënave. Sistemi bankar është i bazuar në sistemin AS400, MIS është një risi për këtë sistem që kombinon modelin e administrimit të të dhënave bankare duke përdorur teknologji të reja. MIS është përdorur me sukses në proçesin e vlerësimit dhe administrimit të kredive te keqija. Kjo platformë përdoret sot në proçesin e hulumtimit paraprak nga banka me reputacion ndërkombëtar si Deutsche Bank apo nga Instituti Financiar Varde Group dhe Centrale Attività Finaziarie (CAF). Qëllimi i këtij studimi nuk është prezantimi i MIS si një produkt, por të paraqesim kompleksitetin dhe fuqinë e këtij aplikacioni dhe se teknikat e përdorura kanë rezultuar të suksesshme dhe ne dëshirojmë të ndajnë këtë me lexuesit për qëllime udhëzuese në aplikime të krahasueshme. I besueshëm dhe i fuqishëm, MIS mund të mbajë miliona të dhëna, dhe është plotësisht i përshkallëzuar në kërkesa, mund të importoj dhe eksportoj të dhëna në formate te ndryshme, pra mund të mbulojë të gjitha aktivitetet monitoruese së bashku me përpunimin, si dhe analizën statistikore.

Në ndërtimin e këtij aplikacioni është përdorur platforma LAMP [9] ku komponentët bazë të kësaj platforme janë:

- GNU/Linux: Sistemi operativ
- Apache: Web serveri
- MySQL: Database server

- PHP: Gjuha skript (Scripting language)

Në këtë aplikacion përveç përdorimit të kësaj platforme të avancuar, janë përdorur dhe teknologjitë më të avancuara të programimit në WEB si XHTML, AJAX, etj.

Për të garantuar vazhdueshmërin e shërbimit dhe për të patur një nivel cilësor të tij serveri është në webfarm Aruba (www.aruba.it). Në fig. 1 paraqitet skema e përdorur për realizimin e testimit.

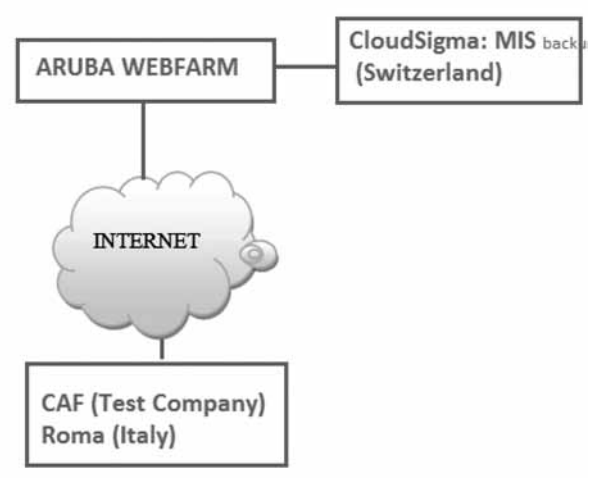

Fig. 1. Skema e përdorur për testimin

3.1 Teknika Cache

Në këtë seksion do të përshkruajmë përdorimin e teknikës Query Cache, duke paraqitur të dhënat reale për të dy gjendjet e sistemit (me Query Cache aktive dhe jo aktive). Në mënyrë që të kemi një krahasim real të rezultateve, matjet janë realizuar në të njëjtën linjë interneti dhe në të njëjtin browser (Google Chrome), me të njëjtët përdorues të lidhur. MySQL përdoret si backend për ruajtjen dhe menaxhimin e përmbajtjes së programit, ku query më i përdorur është tipi SELECT dhe funksionet e leximit, përditësimit dhe fshirjes.

Në MIS nuk është përdorur asnjë lloj teknike për të fshehur të dhënat, kështu në bazën e të dhënave MySQL drejtohen query sa herë që një përdorues bën një kërkesë në faqe. Në situatën kur MIS administron një numër të vogël të të dhënave dhe të përdoruesve, kemi performancë të lartë. Nëse numri i përdoruesve dhe të dhënave rritet, ne kemi vendosur të përdorim teknikën Query Cache e cila realizon ruajtjen e të dhënave direkt në server MySQL. Query Cache është shumë e thjeshtë për t'u përdorur dhe mbi të gjitha nuk ka nevojë për ndonjë ndryshim të kodit PHP. Query Cache ruan outputin e pyetjes së parë SELECT brenda hapësirës së rezervuar për ruajtjen e pyetjeve në RAM, por theksojmë se vetëm query SELECT mund të ruhet. Ky funksion ndihmon përdorimit e të njëjtit rezultat për shumë përdorues, aplikimi kërkon nëse rezultati i kërkuar është në cache. Nëse gjendet, shfaqet në çast, ose query do të ekzekutohet. Gjenerimi i cache ndodh në bazë të kritikës dhe frekuencës së ndryshim të të dhënave. Kjo teknikë mund të përdoret në zona te caktuara ose në të gjithe aplikacionin. Skedulimi dhe aktivizimi për zona të ndryshme nuk është i njëjtë për të gjithë aplikacionin, por janë të vendosur nga analiza e departamentit të IT dhe përvoja e përdorueseve. Duke përdorur këtë teknikë, shmanget realizimi i të njëjtës pyetje në të njëjtën tabelë dhe minimizohen operacionet e I/O në disk. Në rastin e ekzekutimit të query që modifikojnë përmbajtjen e tabelës (UPDATE, INSERT, DELETE, ALTER, etj) të gjitha pyetjet e ruajtura do të fshihen për të lejuar përditësimin e rradhës. Cache Query ndihmon server për të menaxhuar ngarkesën e të dhënave pa u kujdesur për vlefshmërin e të dhënave. Cache Query është një funksion i ofruar nga MySQL database. Fillimisht kontrollojmë nëse Query Cache është aktive:

mysql> SHOW VARIABLES LIKE 'have_query_cache';

Kur query_cache_size i përcaktohet një vlerë jozero, cache query ka nevojë për një madhësi minimale prej rreth 40KB për të alokuar strukturat e saj. (Madhësia e saktë varet nga arkitektura e sistemit).

Aktivizimi bëhet i mundur nëpërmjet kësaj komande:

mysql> SET GLOBAL query_cache_size = XXXXXXX ;

Më poshtë paraqesim një pjesë të kodit të përdorur në procedurë që administron:

QueryCacheManager.class.php

……..

```
function getCacheTableForSql($dp,$sql,$index_fi
elds=array(),$seconds=60,$engine='MyISAM') {
self::purgeCache(getFresh()-
>userSetting('FRESH_SKIP_QUERYCACHE')==1);
$db = $dp->getDbName();
$cacheDp = self::_getCacheDp();
$cacheDb = $cacheDp->getDbName();
$cacheTableDb = self::_getCacheTableDb();
$id = sprintf("%u", crc32(md5($sql).md5($db)));
if($cacheDp->fetchSQLValue("SELECT
```

COUNT(id) FROM fresh_query_cache WHERE id=$id")==1) {return "`$cacheTableDb`.".self::_ getCacheName($id);}
//table does not yet exist: create and populate id self::_insertCache($dp,$id,$sql,$seconds,$engine ,$index_fields);
return "`$cacheTableDb`.".self::_ getCacheName($id);

Në funksion deklarohen:
$dp---------- provider data
$sql ---------- Query
$index_fields=array() ---------- array fileds that will be indexed after the execution
$seconds=60 lifetime of the Cache
$engine='MyISAM' ---------- storage type that is used

4. Rezultatet

Në këtë seksion paraqesim rezultatet e përftuara nga testimet e kryera. Kemi realizuar dy testime, njëherë me Query Cache të aktivizuar dhe njëherë të çaktivizuar. Kemi përdorur dy database një me rreth 500 rekorde dhe një me 200.000 rekorde. Tabela 1 paraqet kohën e ekzekutimit në disa zona të aplikacionit duke përdorur 500 rekorde dhe në të dy gjendjet, Query Cache aktive dhe jo aktive. Në tabelën 2 paraqesim rezultatet e nxjerra nga testimi i kryer me 200000 rekorde. Të paraqitura grafikisht rezultatet jane dhënë në fig. 2 dhe fig. 3.

Tabelë 1: Koha e ekzekutimit në sekonda në disa zona të aplikacionit

Zona	Cache (aktive)	Cache (Jo aktive)
FRMWarnings	0.0063	0.0076
TopHeader	0.085	0.0923
ChooseNDG	0.1534	0.1861
ActivityTimeline	1.393	3.9213
Home	0.1693	1.3234
Notifications	0.1271	0.5766

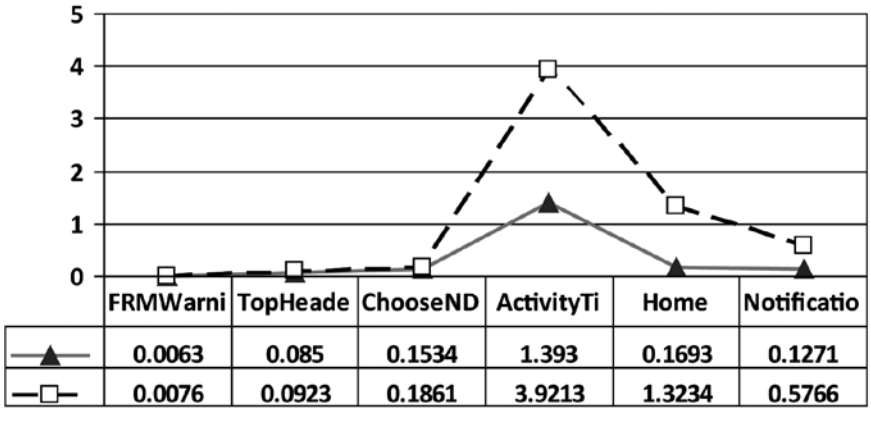

Fig. 2: Paraqitja grafike e rezultateve të testimit për 500 rekorde

Tabelë 2: Koha e ekzekutimit në sekonda në disa zona të aplikacionit për 200000 rekorde

Zona	Cache (aktive)	Cache (jo aktive)
FRMWarnings	0.0046	0.0082
TopHeader	0.0591	0.1252
ChooseNDG	0.1613	0.3032
ActivityTimeline	14.2095	44.6418
Home	3.2402	0.1447
NDGedit	0.292	1.8744
BorrowerResultsLister	2.7635	7.821

Zona	Cache (aktive)	Cache (jo aktive)
ResolutionsStrats	34.506	96.053
BpStrats	0.6014	2.4047
FRMTaskLister	1.3607	4.7414
Notifications	0.1583	5.7257
Watchlist	0.8517	2.7389
PropertiesLister	0.0379	5.8394

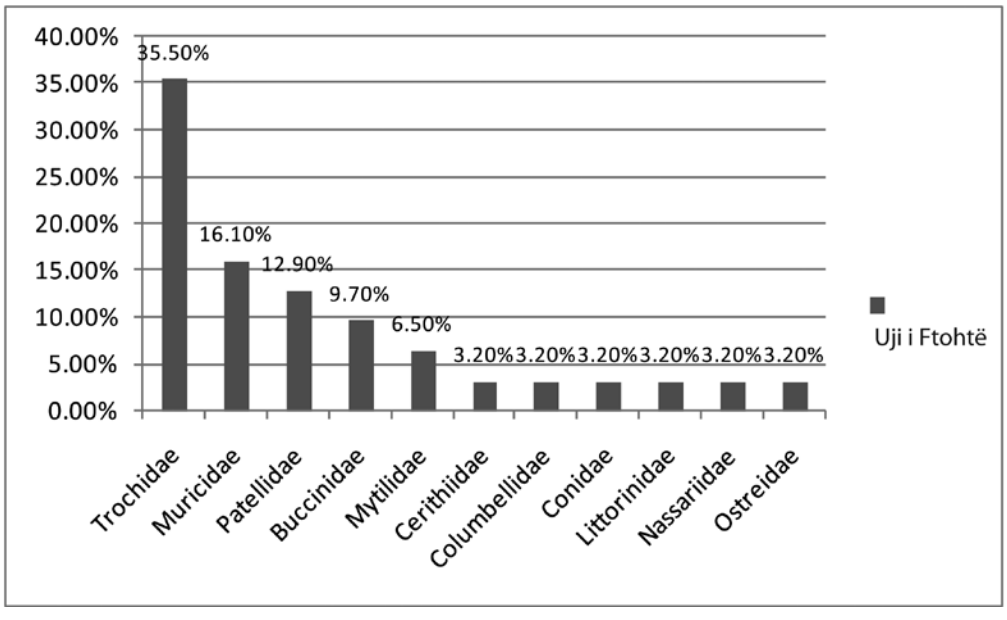

Fig. 3: Paraqitja grafike e rezultateve të testimit për 200000 rekorde

5. Konkluzione dhe çështje për të ardhmen

Duke u bazuar në eskperiencën e gjatë në projektimin dhe administrimin e aplikacioneve web dhe në rezultatet e matjeve të bëra nxjerrim si konkluzion që QueryCache është një teknikë e dobishme dhe e këshillueshme për tu përdorur në të gjitha aplikacionet web ku si back-end përdoret MySQL. Kjo teknikë përmirëson performancën e aplikacionit në masën 200 %. Nga matjet e kryera rezulton se në seksionet me më shumë Query për tu ekzekutuar ka një ndryshim të ndjeshëm mes aktivizimit dhe mos aktivizimit të QueryCache, ndërsa në ato me më pak Query ndryshimi është i vogël. Këshillohet që për aktivizimin e kësaj teknike të përdoret një konfigurim i personalizuar dhe jo duke përdorur konfigurimin default të saj. Si punë të rradhës do të trajtojmë një teknikë tjetër të përmirësimit të performancës, e cila është Indeksimi.

Falenderime

Duam të falenderojmë Centrale Attività Finanziarie S.P.A (Rome), Misoft S.R.L (Rome), bashkëpunëtorin Pietro Baricco për ndihmën që na ofruan në përdorimin e serverave, aplikacionit etj.

Referenca

[1] Liu, H. (2009). Software performance and scalability a quantitative approach, Wiley series on Quantitative Software Engineering.

[2] Jim Challenger, Arun Iyengar, and Paul Dantzig. A Scalable System for Consis-tently Caching Dynamic Web Data. Proc. IEEE INFOCOM 99.

[3] MySQL AB. MySQL: The World's Most Popular Open Source Database. www.mysql.org. 2005.

[4] Murugesan, S. "Understanding Web 2.0," IT Professional , vol.9, no.4, pp.34-41, July-Aug. 2007

[5] J. Garrett, ―AJAX: A new approach to Web applications‖, Adaptive Path, 2005

[6] ASP.NET Ajax: http://www.asp.net/ajax

[7] Meli , S.; G mez, J.; P rez, S.; D az, O.; , "Architectural and Technological Variability in Rich Internet Applications," Internet Computing, IEEE , vol.14, no.3, pp.24-32, May-June 2010

[8] G.A. Di Lucca, A.R. Fasolino, ―Testing Web-Based Applications: the State of the Art and Future Trends‖, Information and Software Technology Journal, Vol. 48, Issue 12, Pages: 1172-1186 (December 2006), Elsevier inc.

[9] Dougherty D., LAMP: The Open Source Web Platform. www.onlamp.com/pub/a/onlamp/2001/0 1/25/lamp.html, 2001 as accessed on May 2011.

[10] Fraternali, Piero; Rossi, Gustavo; Sánchez-Figueroa, Fernando; , "Rich Internet Applications," Internet Computing, IEEE , vol.14, no.3, pp.9-12, May-June 2010