O. O. Kovalyuk, Cand. Sc. (Eng.), Assist. Prof.

ANALYSIS OF SOFTWARE FOR CACHING DATA IN HIGH-LOADED SYSTEMS

The paper studies data caching software for optimizing operation of high-loaded web systems and substantiates the choice of caching means depending on data volumes.

Keywords: *caching, high-loaded systems, web site, optimization.*

Introduction

Currently, a considerable number of computer systems are created in the form of web applications where a web browser acts as a client interacting with a web server. Such architecture is implemented in many computer control systems, business process control systems, web sites. The main advantage of such approach is the use of a "thin client" (a browser), which excludes the necessity of an additional software installation for a user. One of the main problems arising during the web system operation is to provide their interaction. The most common way of solving this task is data caching, i.e. locating data in an intermediate buffer in order to provide a quicker access to it.

There are various approaches to data caching and different ways of their implementation [1].

According to the mechanics of the work, the following types of caching are distinguished [2]:

– lazy cache – stores data and returns it until the cash becomes outdated

– synchronized cache –the client receives a mark together with data and in the following access asks if the data has changed in order not to receive it once again. Such approach is realized by http-protocol.

– write-through cache – any change in the data takes pace in the source and in the cache simultaneously.

According to the type of data we may distinguish:

– caching data that are presented to a user (front-end);

– caching data that are used for forming html-markup (back-end).

According to cache location, there are the following types of caching (Fig. 1):

– caching on the client side (a local cache of the browser);

– local cache of the web server (used for caching static content);

– caching by the application server.

Though caching is widely used, there are no common rules to choose a caching system [3, 4]. As a rule, the choice of caching means is conducted by a developer on the basis of personal experience and preferences.

Therefore, the task of studying the efficiency of caching means for different data volumes is of current importance.

The paper aims at obtaining the dependencies of writing / reading time on the amount of data in the cache and finding the optimal caching means on the basis of experimental studies.

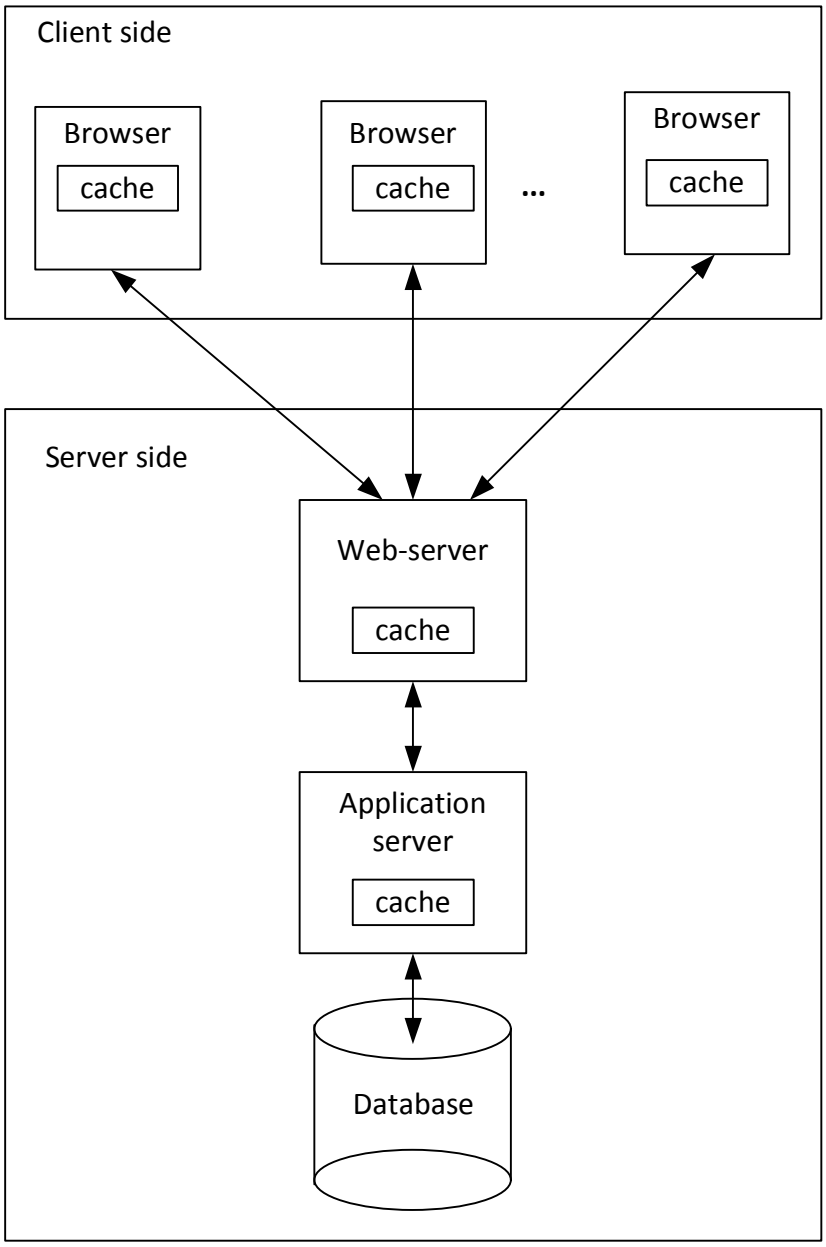

Fig. 1. Possible location of caching means

Procedure of performing experimental studies

In this study the analysis of caching the results of SQL requests is conducted by the example of internet site www.zabudovnyk.com.ua — a highly loaded real estate portal. Caching the results of SQL queries is determined by the fact that addressing a database is a "bottleneck" of many web systems. The portal was created with the application of Zend Framework 1 library, and so a software with the support implemented in Zend Framework will be tested as caching means:

- APC;
- Memcached;
- File cache.

Experimental procedure includes the following steps:

1. Obtaining experimental data about the speed of writing data to cache for different data volumes. At this step data is chosen from a database, a key for data identification in the cache is formed, the data is written to the cache. The time of placing data into the cache is considered as the time of running the function which saves data in the cache.

2. Obtaining experimental data about the speed of *reading* data from the cache for the data placed into the cache at step 1.
3. Visualization of the data.

Experimental conditions:

- The studies are performed using a test server (Linux Fedora, SSD disks);
- The time of data deletion from the cache is not investigated;
- It is assumed that there is data in the cache.
- To achieve more accurate results, several modeling iterations are used and averaged results are determined.

In order to provide flexible adjustment of the system and simplify the research, automation of caching means selection has been performed. The type of caching means is specified in the configuration file of the system and switching between caching means does not require any changes in the program code.

Experimental research results

The results of writing data to the cache are presented in Table 1 and Fig. 2.

Table 1

The time of writing data to the cache, s.

The amount of data / Cache	500	5000	50000	500000
APC	0.046	0.46	4.5	45
File	1.8853	70.0000	-	-
Memcached	0.053	0.53	5.3	53

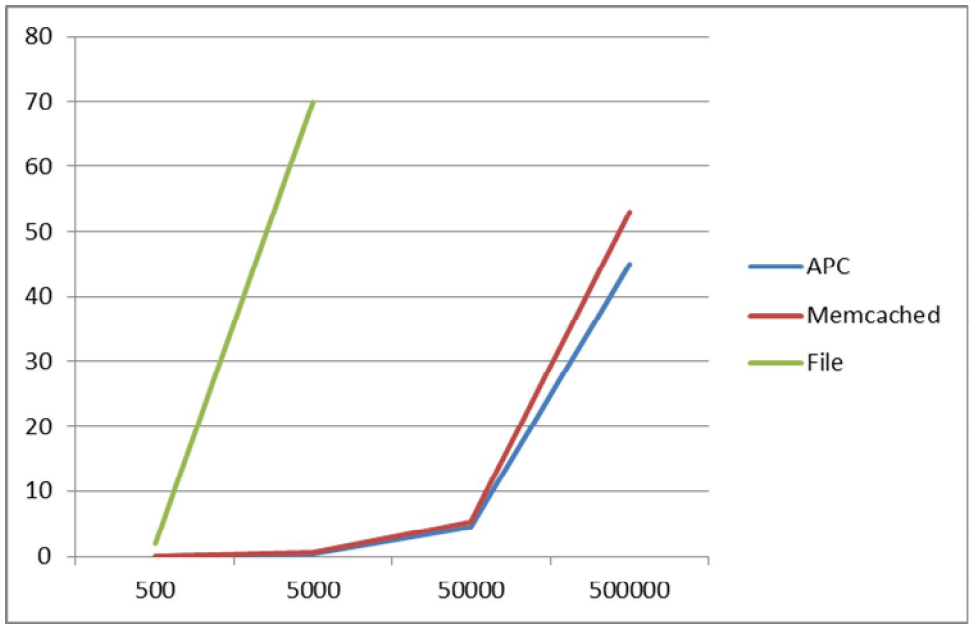

Fig. 2. The results of writing to the cache

The results of reading data from the cache are presented in Table 2 and Fig. 3.

Table 2

The results of reading data from the cache, s

The amount of data / Cache	500	5000	50000	500000
APC	0.0108	0.1089	0.6809	13.3850
File	0.0932	1.0519	-	-
Memcache	0.0178	0.1760	1.7935	18.0930

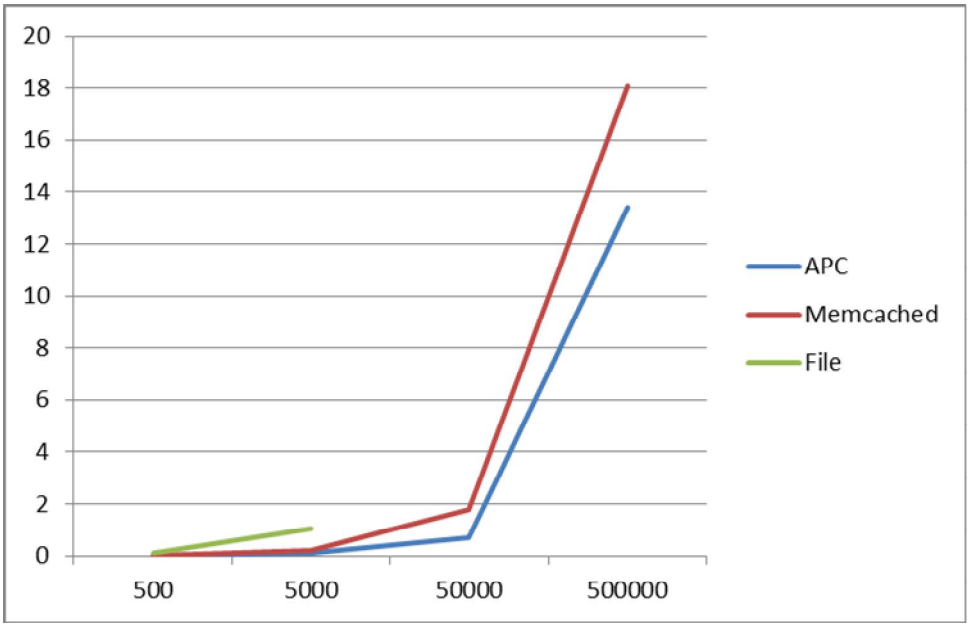

Fig. 3. The results of reading data from the cache

Conclusions

On the basis of caching means investigation the following conclusions can be made:

– Means for caching data in read / write memory (APC, Memcached) are much faster than file cache.

– It is inexpedient to use file cache for large amounts of data since each value is saved in a separate file. For this reason testing of the file cache with the number of iterations exceeding 5000 was interrupted due to a long running time.

– Writing time exceeds reading time. For APC and Memcached there is a several time excess and for a file cache there is a several dozen excess.

– For APC and Memcached writing time is practically nondependent on the amount of data that are already in the cache.

– For APC and Memcache reading time increases insignificantly with the increased amount of data in the cache.

Thus, the procedure of caching means selection can be formulated as that including the following steps:

1. To choose software means that uses read-write memory for data storage as the main cache. Preference is given to APC.

2. If the resources of read-write memory are limited, a file cache could be additionally used for caching up to 5000 entities.

3. If APC is used, application of the integrated web interface will be expedient for monitoring the main indicators of cache utilization (Fig. 4).

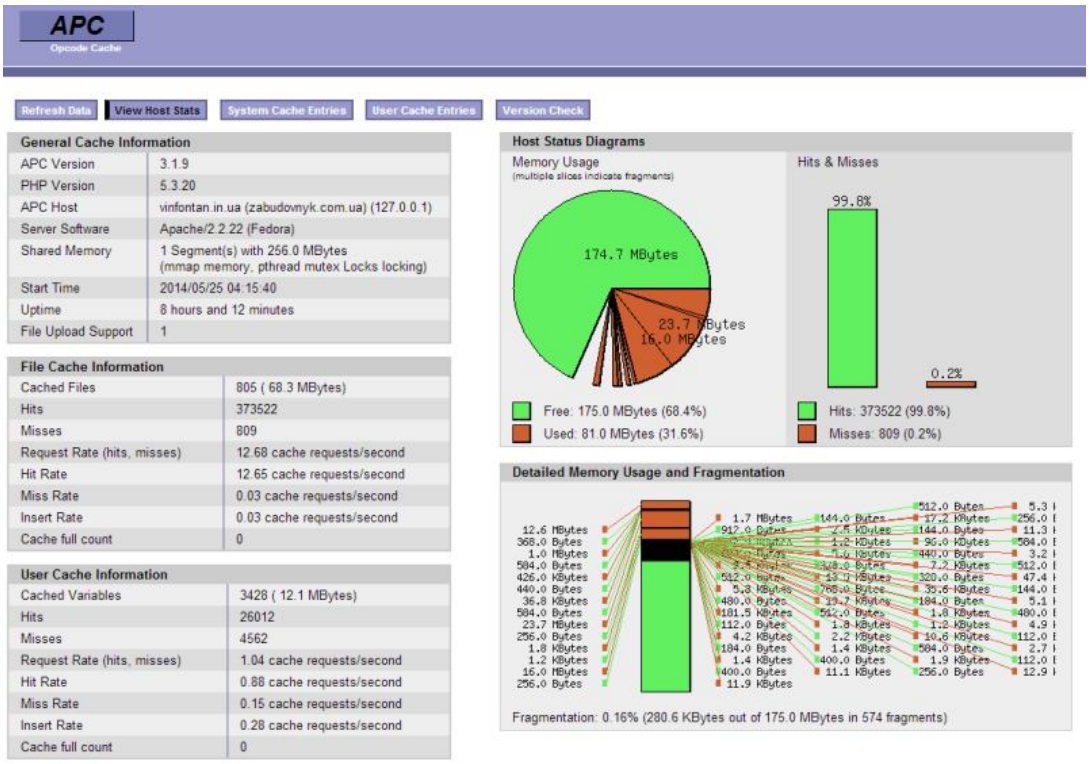

Fig. 4. Web-unterface of APC

REFERENCES

1. Ботыгин И. А. Исследование методов увеличения производительности web-приложений // И. А. Ботыгин, К. А. Каликин / Известия Томского политехнического университета. – 2008. – Т. 312, № 5. – С. 109 – 114.

2. Стас Выщепан Стратегия кеширования в приложении [Електронний ресурс] // Режим доступу : http://habrahabr.ru/post/168725.

3. Теория кэша (часть вторая, практическая, дополненная) [Електронний ресурс] // Режим доступу : http://habrahabr.ru/post/38911/.

4. Michal Špaček Caching Strategies [Електронний ресурс] // Режим доступу : http://www.slideshare.net/spaze/caching-strategies.

Kovalyuk Oleh – Cand. Sc. (Eng.) Ass. Prof. of the Department of Computer Control Systems.
Vinnytsia National Technical University.