A STUDY ON MOODLE'S PERFORMACE

José Coelho, Vitor Rocio, Universidade Aberta, Portugal

1. Introduction

Learning Management Systems (LMS) are essential tools to the modern teaching institutions. Moodle[1] is an open source LMS, widely used by open and distance teaching universities, as well as support to face to face courses. There are almost 40,000 registered Moodle sites, all over the world. Moodle can be installed in a wide range of environments: operating systems (Linux, Windows), supporting databases (MySQL, PostgreSQL, ...), and hardware.

This paper addresses the issue of Moodle's performance within different environments, and under different loads. What is worth changing and what isn't to improve performance? When load increases, what should be changed in order to obtain the largest performance gain?

To answer these questions, it is important to subject the application, in a test environment, to the real conditions of use. In order to do it, we use historical data on the use of Moodle in Universidade Aberta (UAb). Several Moodle instances will be subject to different load levels and the resulting performance will be measured. It is difficult to estimate instantaneous load, even in a working site, where users' habits are known. The number of users is an easier question to address and may be estimated by the number of teachers and students that will be using the LMS. So, in order to estimate future load, we present a conversion method according to the present level of use in this university.

The rest of the text is organized as follows: in section 2 we explain our option on the performance indicator, and enumerate a number of factors that it depends on. In section 3 we propose a test environment and select a subset of factors to study, and in section 4 we address the question of the estimation of server load. In section 5 we describe the tests done and results obtained, and we finish in section 6 with some conclusions and future work. The used SQL queries are presented in annex.

2. Performance

As performance indicator, we use the mean loading time of a web page in the site, including all images and embedded objects. We chose this indicator mainly because it's directly related to the user's experience when navigating the site. If loading times are long, there is a degradation of users' experiences, and in peak times, people typically tend to blame the software.

On the other hand, if pages are consistently quick to load, users acquire the necessary confidence to use the application. This confidence is essential for users to become active, to routinely use the LMS, and contributing to the success of online teaching and learning.

The LMS server must therefore have a good performance at all times (reliability and availability [1]), so we need to know what factors performance depends on. Our experience points to the following ones:

- Downloaded page size, with embedded images and objects;
- Available bandwidth;
- Server load, at the moment the page is requested;
- Available memory;
- Processing capacity (CPU);
- Operating system and respective version;

[1] http://moodle.org

- Moodle version;
- Database management system and respective version;
- Web server and respective version;
- Database size;
- Server architecture.

These factors, among others, influence the application's performance, in spite of belonging to different categories. Some can be estimated for a specific situation, others are decision variables. In the next section, we will describe a test environment that can be used and/or adapted to each institution's reality.

3. Test environment

The test we propose consists on the measurement of mean response time of a set of selected pages, during 5 minutes. In order to emulate production conditions, the number of users and average time between mouse clicks varies, so that we can measure performance for different use loads. The test is repeated for a set of instances with different installation options, in order to obtain the impact of those options in the application's final performance.

Download size is a factor that users are easily aware of: large files aren't generally embedded in the page, and the browser usually shows download progress. So, pages without large embedded files are expected to load quickly. In this test, a set of different Moodle pages is considered, in order to cover a normal use. Large file access is not covered, since users will understand the delay in those cases. Pages of the following types: main page; login page; password recovery; course's main page; forum view; topic view; glossary; resource files; database activity. Pages are randomly chosen during the test, with equal probability.

Bandwidth can limit server access, once load and page size surpasses the allowed bandwidth. If the limit is not achieved, it doesn't affect the test, so we will assume we have a large enough bandwidth and will not consider it.

The key factor is server load. Load can be measured by number of pages requested per minute. A Moodle user doesn't know what load the system has, and even for an administrator, it may be difficult to predict behaviour in future peak uses, unless he performs tests under extreme conditions. Since this parameter is determinant in performance, we propose in the next section, a method to estimate load through available historical use data.

The other factors are installation options. There are many, differing in cost and complexity. A thorough evaluation of these options is not the goal of this paper, we intend only to give some indications on the most relevant factors when choosing an architecture (software and hardware) to run Moodle.

Studied factors are as follows:

- Database dimension: small (3 MB) vs. large (400 MB)
- Moodle version: 1.7 vs. 1.8 vs. 1.9
- DBMS: MySQL vs. PostgreSQL[2]
- Server architecture: single server vs. 2 servers (front end/back end)
- Operating system/hardware: Windows-based amateur PC[3]; Linux-based professional server[4]

The questions we intend to answer are: How does performance degrade with the increase of the database? What is the performance difference among different versions? Is there a performance difference in DBMS? How much and in what load conditions? What is the gain by having two servers instead of one? What is the performance boost when using a professional server architecture when compared to a low end system?

In section 5 we describe a series of tests that start to answer these questions. To perform these tests, we need to know the real load level that the system is subject to, and this is the purpose of the following section.

---

[2] http://www.mysql.com; http://www.postgresql.org

[3] Dell PowerEdge 750, Pentium 4 2.8GHz, 512MB RAM, Windows Server 2003

[4] Acer Altos R510Mk2, Xeon 4x3.0GHz, 4GB RAM, Fedora 8

4. Test load

Server load can be measured at each moment by counting the number of page views per minute, that naturally varies through the day. Our goal is to estimate its peak value, since it corresponds to the moment when the server is subject to a larger load, and more users are connected.

There are two ways to obtain statistical information on the server load: by analysing the web server's log files, or by querying Moodle's mdl_log database table, that registers all events in the LMS. These two sources of information dot not coincide, one complements the other. For instance, the number of page views in October 2007 in UAb is 2,201,980, according to the database, and 4,874,261 according to the log files. In what value should we believe? The answer becomes clear when we analyse the URLs of the top pages accessed: log files account for auxiliary pages (stylesheets, scripts, ...) while the Moodle database doesn't. The information in the database is naturally more reliable, since the application details are taken into account. The web server is blind with regard to the application it is running. Therefore, we choose to use the database info whenever possible and use the log files info as complementary data. All database queries are shown in the annex, so that readers can repeat the analysis in this paper.

There is a large difference between a casual user that registers in Moodle out of curiosity, but rarely uses it afterwards, and a user that connects to the platform in a daily basis. In order to determine what kind of user explains the number of page views, we considered: the number of page views per month, from October 2005 to December 2007; the number of users that accessed the platform at least once in each month; and the number of active users. Active users are defined as those who access the platform at least on 15 different days in each month. As an alternative we considered also those users who access the platform at least on 10 and 5 different days in each month. Figure 1 shows a scatter plot with linear adjustments, for this data.

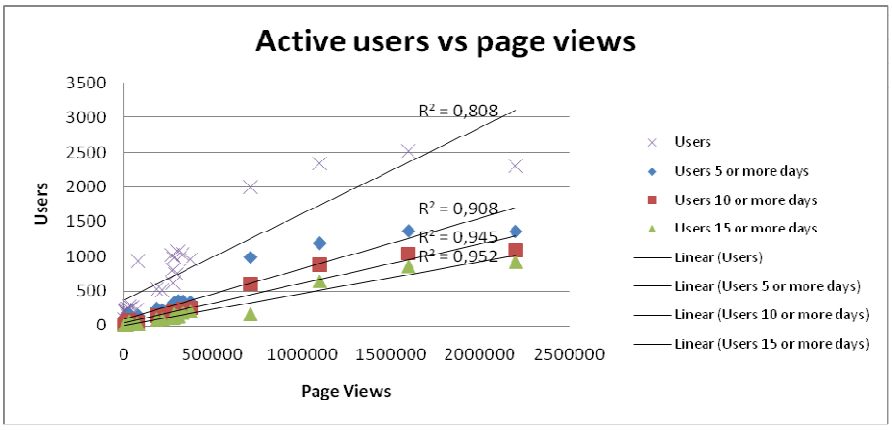

Figure 1 – Active users vs. page views

As we can see, active users that accessed at least 15 days, are the ones that best explain page views, even if the adjustment is not perfect, probably being very dependent on the online methodology used in the institution. In UAb, adjustment for this period is `Page views per month = active users * 1758`. We will adopt a pessimistic approach for this and upcoming expressions, so we'll round constants to values that represent a worse scenario: `Page views per month = active users * 2000.`

After estimating page views per month, we'll need to estimate the maximum number of page views per day, and we can do better than to simply divide by the number of days in the month, since server load varies along the week. Table 1 shows the total number of page views per week day in UAb's Moodle, since it started being used, and their respective percentages. As we can see, Monday is the most loaded day, about 18% of the whole 7-day week, being 1.24 times more than the average. This factor can be used to correct the number of page views per day, when converting from the number of page views per month: we multiply the number of page views per month by 1.24/30 = 4.1%, which, when applying our pessimistic criteria, becomes 5%.

Table 1 - Total number of page views per week day

Week day	Page views	%
Sunday	1023126	10,6%
Monday	1719622	17,8%
Tuesday	1624534	16,8%
Wednesday	1549641	16,0%
Thursday	1515350	15,7%
Friday	1331909	13,8%
Saturday	908703	9,4%

To find out the peak load, we need to study traffic distribution throughout the day, in order to determine the most loaded hour. Figure 2 shows traffic distribution throughout the 24 hours of the day.

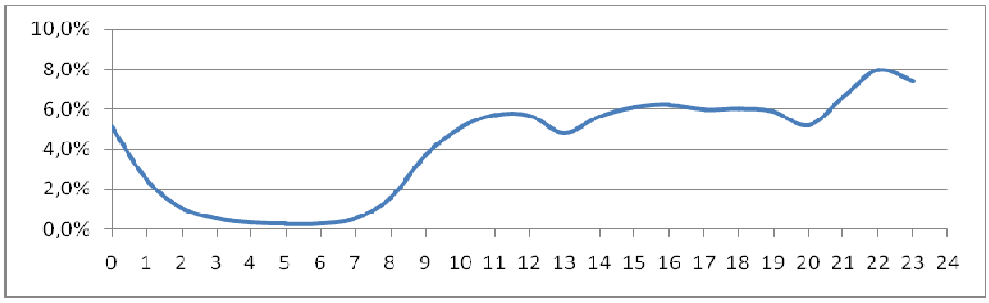

Figure 2 – Traffic distribution throughout the 24h

As we can see, the peak is achieved at 22:00, with 8% of all pages viewed in a day. Again, applying a pessimistic factor, we round this value to 10%.

Thus, the estimated value of page views per hour during peak times is equal to page views per day divided by 10.

```
Page views per month = active users * 2000
Page views per day = page views per month / 20
Page views per hour = page views per day / 10
Page views per minute = page views per hour / 60
```

5. Tests and results

The Moodle instances considered are intented to cover the vectors discussed in section 2. In table 2 we show the instances that were tested.

Table 2 – Tested Moodle instances

Instance ID	Moodle version	Database dimension	DBMS	Servers	OS/Hardware
UAb1	1.7	3 MB	MySQL	1	Linux/Professional
UAb2	1.7	400 MB	MySQL	1	Linux/Professional
UAb3	1.8	3 MB	MySQL	1	Linux/Professional
UAb4	1.9	3 MB	MySQL	1	Linux/Professional
UAb5	1.9	3 MB	MySQL	2	Linux/Professional
UAb6	1.9	3 MB	PostgreSQL	1	Linux/Professional
UAb7	1.9	3 MB	MySQL	1	Windows/Amateur
UAb8	1.7	3 MB	MySQL	2	Linux/Professional
UAb9	1.9	400 MB	MySQL	1	Linux/Professional
UAb10	1.9	400 MB	MySQL	2	Linux/Professional

Each Moodle instance is submitted to different loads during 5 minutes, using Webstress Tool[5]. All machines involved in the test are connected to a 1 Gb/s network, so we can disregard this factor.

The number of users and average time between mouse clicks must be specified for each configuration. We see that we obtain different performance values for two configurations with the same number of page views per minute, but with different number of connected users or time between mouse clicks. As a consequence, it is essential to estimate the number of users during the test period in order to approximate the test to reality.

A query to the database allows us to obtain the number of users in 5-minute intervals, and the corresponding average number of page views. In this way, we can obtain the average of page views per user during the test interval, and also the average time between mouse clicks. While the number of users varies widely, from 1 to 120 users in 5 minutes, the time between mouse clicks is rather stable, with an average of 79 seconds. With a pessimistic perspective, we will round this number to 60 seconds, and vary the number of users in order to vary the use load. The tested loads are described in table 3.

Table 3 – Tested loads

Number of users / click delay	Load	pv/minute	pv/hour	pv/day	pv/month	active users/month
12 / 60	lowest	12	720	7200	144000	72
30 / 60	low	30	1800	18000	360000	180
60 / 60	medium	60	3600	36000	720000	360
120 / 60	high	120	7200	72000	1440000	720
300 / 60	peak	300	18000	180000	3600000	1800

The performance results obtained, measured in miliseconds, are presented in table 4.

Table 4 – Average response times

Instance	lowest	low	medium	high	peak
UAb1	610	746	1055	2460	26640
UAb2	923	1029	1771	14411	42414
UAb3	765	1011	1549	5487	33126
UAb4	688	837	1066	3924	30756
UAb5	647	844	1306	3661	30003
UAb6	708	1024	1405	5880	34739
UAb7	1656	5271	26986	98910	-
UAb8	1037	1302	2259	6982	35918
UAb9	920	1233	2131	14373	43084
UAb10	1062	1305	2214	12473	39589

To better analyse the differences of performace, table 5 shows increases in response times between configuration pairs, grouped by each studied factor.

Table 5 – Comparison of the performance of instances

DB Dimension (3MB->400MB)	none	low	medium	high	peak
UAb1 ->UAb2, Moodle 1.7	51%	38%	68%	486%	59%
UAb4 -> UAb9, Moodle 1.9	34%	47%	100%	266%	40%
UAb5 - >UAb10, Moodle 1.9, 2 servers	64%	55%	69%	241%	32%

[5] http://www.paessler.com

Moodle version					
UAb1 -> UAb3 (1.7->1.8)	25%	35%	47%	123%	24%
UAb3 -> UAb4 (1.8->1.9)	-10%	-17%	-31%	-28%	-7%
UAb1 -> UAb4 (1.7->1.9)	13%	12%	1%	60%	15%
UAb2 -> UAb9, DB 400MB (1.7->1.9)	0%	20%	20%	0%	2%
UAb8 -> UAb10, DB 400MB, 2 servers (1.7->1.9)	2%	0%	-2%	79%	10%
DBMS (MySQL -> PostgreSQL)					
UAb4 -> UAb6	3%	22%	32%	50%	13%
Architecture (1 server -> 2 servers)					
UAb4 -> UAb5, Moodle 1.9	-6%	1%	22%	-7%	-2%
UAb2 -> UAb8, Moodle 1.7, DB 400MB	12%	26%	28%	-52%	-15%
UAb9 -> UAb10, Moodle 1.9, DB 400MB	16%	6%	4%	-13%	-8%
Hardware/OS (professional -> amateur)					
Uab4 -> Uab7	141%	530%	2431%	2421%	

We can observe a considerable increase in average response time when the database size increases. The difference is larger in high load, where response time is still acceptable for small databases, but prohibitive for large databases. The tests clearly reveal that version 1.8 is slower than 1.7 or 1.9, all other factors being the same. We can also see that PostgreSQL is slower to respond than MySQL. The increase in response time is larger for heavier loads. The tests on the architectural differences seem to indicate that 2 servers can improve response time for higher loads, but communication overhead is a factor in lower loads. Finally, and not surprisingly, the use of high-end hardware is the factor that most dramatically improves performace.

6. Conclusions and future work

In this paper, we proposed a method for testing the impact of installation options in the performance of Moodle. Also, a method for estimating the peak load was proposed, that is adaptable to each institution. Finally, we study the impact on performance of the database dimension, Moodle version, database type, architecture and hardware/OS. We concluded that high-end hardware and small database size are the most important factors that contribute to performance. Since data tends to increase, systems administrators are advised to consider periodical archival of old data and database optimization. As future work we point in several directions: test clusters with more servers; study the database growth with site usage; consider more database types, more operating systems, more web server types.

References

1. TANENBAUM, A.; VAN STEEN, M. (2007). Distributed Operating Systems (2nd Edition). Prentice-Hall.

Annex - Queries

PageViews / Hour	`SELECT hour( FROM_UNIXTIME( time ) ) AS HORA, Count( * ) AS "Paginas Vistas" FROM mdl_log GROUP BY HORA ORDER BY HORA;`
Page Views / DayOfWeek	`SELECT dayofweek( FROM_UNIXTIME( time ) ) AS dayofweek, Count( * ) AS "Paginas Vistas" FROM mdl_log GROUP BY dayofweek ORDER BY dayofweek;`
PageViews / Month	`SELECT month( FROM_UNIXTIME( time ) ) AS mes, year( FROM_UNIXTIME( time ) ) as Ano, Count( * ) AS "Paginas Vistas" FROM mdl_log GROUP BY ano,mes ORDER BY ano,mes;`
Users / Month	`SELECT Mes, Ano, Count(*) As "Utilizadores" FROM (SELECT userid, month( FROM_UNIXTIME( time ) ) AS Mes, year( FROM_UNIXTIME( time ) ) As Ano FROM mdl_log GROUP BY userid, Mes, Ano) As USR GROUP BY ano, mes ORDER BY ano, mes;`
ActiveUsers15 / Month	`SELECT Mes, Ano, Count(*) As "Utilizadores" FROM (SELECT userid, Mes, Ano, Count(*) As "Dias" FROM (SELECT userid, day(FROM_UNIXTIME(time)) AS Dia, month(FROM_UNIXTIME(time)) AS Mes, year(FROM_UNIXTIME(time)) As Ano FROM mdl_log GROUP BY userid, Dia, Mes, Ano) As USR GROUP BY userid, ano, mes HAVING Dias>14) As USR2 GROUP BY ano, mes ORDER BY ano, mes;`
Average PageViews by number of users in 5 minutes of the last week	`SELECT Count(*) As 5Minutos, AVG(PaginasVistas) As AvPV, Utilizadores FROM (SELECT SUM(pv) As PaginasVistas, Count(*) as Utilizadores FROM (SELECT Count(*) As pv, userid, minute(FROM_UNIXTIME(time)) DIV 5 AS Minuto, hour(FROM_UNIXTIME(time)) AS hora, day(FROM_UNIXTIME(time)) AS Dia, month(FROM_UNIXTIME(time)) AS Mes, year(FROM_UNIXTIME(time)) As Ano FROM mdl_log WHERE time > UNIX_TIMESTAMP(SUBTIME(Now( ) , '7 0:0:0')) GROUP BY userid, minuto, hora, dia, Mes, Ano) As USR GROUP BY minuto, hora, dia, Mes, Ano) As USR2 GROUP BY Utilizadores ORDER BY Utilizadores;`