Using PHP and HTML languages and MySQL databases to create servers of scientific information

Bogdan Księżopolski[*]

Institute of Physics, Maria Curie-Skłodowska University,
Pl. M.Curie-Skłodowskiej 1, 20-031 Lublin, Poland

Abstract

In the paper we present short profile of different compositions of many-layered applications. We compare speed and stability of the different sets of applications. We describe a configuration which can be used to create a server of scientific information. The application consists of MySQL database management system, http server (Apache) and PHP language. All operate in the Linux environment.

1. Introduction

The huge quantities of information cross through research institutions every day. Then they are exchanged among different scientific teams all over the world, which in a considerable way inspires the development of different branches of science. Each information, which we are using stays partially in our memory but of course it is not worth remembering its whole content. But it frequently happens that information which seems to be unnecessary now, becomes useful in the future. So, it is worth to keeping reaching information and this is one for the the main reasons of using the database systems in science.

2. Choice of components of many-layered applications

There are many ways of using the database systems. The best system, easy to reach is the one based on Internet. The use of many-layered application seems to be an ideal solution of using such a system. Our aim is to connect three elements: the database engine, programming language as well as HTTP server. Environment in which these elements work is also important.

---

[*] *E-mail address*: ksiezop@tytan.umcs.lublin.pl

2.1. Combination of different configurations of many-layered application

Choice of components of this application is the most important decision during creating database systems. The team of French Internet portal [1] have tested several compositions of such applications. We will present the results of this research below.

Tests were taken on the Compaq Proliant server with the Pentim II 400 Mhz processor, 512 MB of RAM memory and three SCSI disks – 4 GB for system and 2x9 GB for data. HTTP servers connect with the client terminals the database servers by using 100 Mb/s full duplex FastEthernet network.

Two combinations of WWW servers and two database servers were prepared to compare the many-layered applications. The first WWW server was constructed on the open source programs – Linux (core 2.4.6, mandrake distribution), apache (1.3.19) and PHP (4.0.4.p1) interpreter. The second WWW server was based on the Microsoft products – Windows 2000 Server with Service Pack 2, Internet Information Server 5 with ODBC drivers to freeware database engines: MySQL (MyODBC 2.50.39) and PostgreSQL (PgODBC 7.01).

Database servers were installed on the same system platforms – on Linux, MySQL 3.23.26 and PostgreSQL 7.03, meanwhile on Windows – MS SQL Server 2000. A workstation with Microsoft Windows NT 4 or Windows 2000 with software Microsoft Web Stress 1.1.293.1 were used to simulate access of up to 150 clients at the same time.

The data used for this test were prepared identically for all database servers and they included 5000 records in the schedule of following structure:

- field integer with the automatically enlarged value, being also the index of the schedule
- text field with the constant length 255 signs
- field with the date and time
- integer value
- logical value
- text field of variable length (maximum 255 signs)

The test has answered several questions, however it is worth mentioning the most essential problems connected with creating scientific servers.

Firstly, the range of records (from 3000 to 3020) in the database was displayed (Fig. 1). Secondly, the random chosen record was displayed (Fig. 2). By analyzing the first case we can came to the conclusion that we get more answers during one second in the Windows applications and particularly in connection with the PostgreSQL database system. The Linux applications have

got also good estimation, however, they were slightly worse than the Windows ones.

If we analyse the stability of individual sets of application, it is possible to claim that two systems with the PosgreSQL database are stable even in sessions with 150 parallel questions. In this test, the applications based on the MySQL database engine lose stability at about 30 parallel questions.

Analyzing the test of searching individual records in the database, we can notice, that the applications based on the Windows environment are less effective than the Linux applications. The most realized demands per second were made by the applications based on the Linux environment, particularly in connection with the MySQL database.
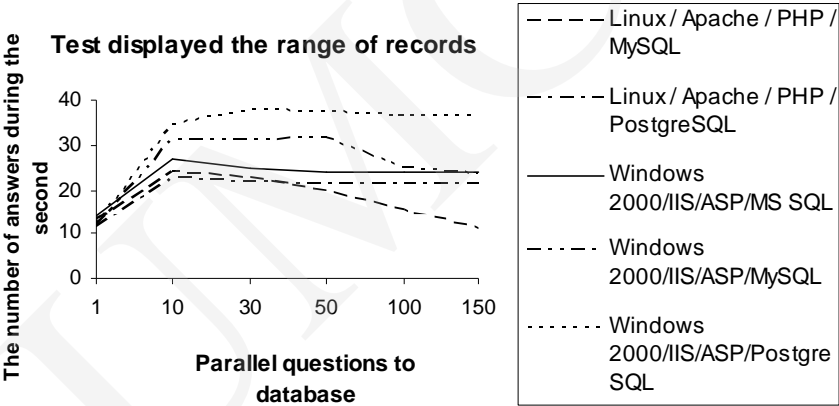

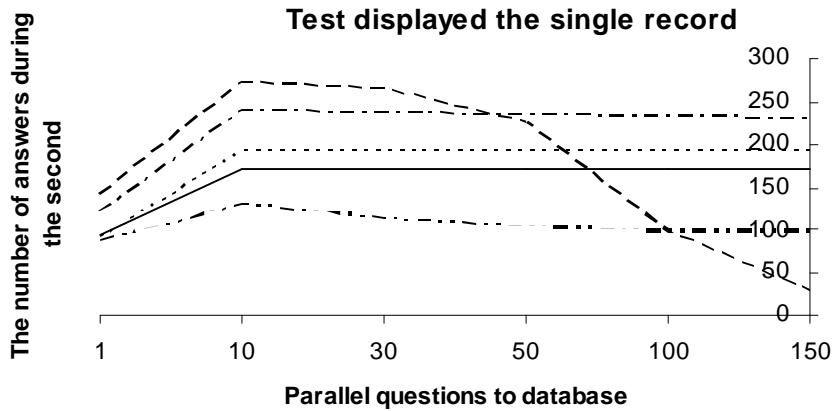Fig. 1. Results of the test displayed the range of records

Fig. 2. Results of the test when a single record is displayed (legend the same as from Fig. 1)

Comparing many-layered applications in terms of stability, we can come to the conclusion that the systems based on the PosgreSQL database are more stable in the case of several parallel questions. The applications based on the MySQL database are very quick, however, they lose their efficiency with over 30 parallel demands to database access.

Summing up, creating the server of scientific information with a large intensity of motion, it is worth choosing the system of PosgreSQL database. If only a few persons are using simultaneously the application, then the system of MySQL database seems to be better solution, which is connected with its speed.

Constructing a many-layered application, we should take into account its economical issue. Components of many-layered applications operating in the Linux environment are worth recommending as the operating system Linux is free. To purchase the software based on the Windows environment it is necessary to allocate a considerable sum of money for the adequate licenses.

Adding up all the above arguments which support particular sets of many - layered applications we can come to the conclusion, that, in the case of servers of scientific information, it is worth using free software. They offer efficiency comparable with the commercial applications but with considerable economic advantages. Choosing the system of management of our database, it is necessary to take MySQL system for servers with comparatively small intensity of parallel demands, however, for servers with large stability of parallel demands it is better to use the PostgreSQL system.

3. Programming the database

Having already chosen the components of many-layered application and the operation system, we can come to the process of programming the database. In the case of server of scientific information, which was programmed in the Department of Theoretical Physics in Lublin, the many-layered application is operating in the Linux environment. This application consists of: system of management of the MySQL database [2], http Apache [3] server and PHP language [4].

The first step is to create the project of database. The reason why it is so important to spend some time on creating of a logical project of the database before access to its implementation, is the fact that this has the key importance for cohesion, integrity, precision of the data kept in the base. If the base is not well thought out, then its users could have problems with reading information; there is also danger that this information will be incomplete or even incorrect.

If concept of the project is already finished, it is necessary to create designed schedules. One has to address this directly to the computer on which the server should be installed and then after getting access to the base one can create relation. One can do this by using the query structural language SQL, writing

down its commands directly from terminal. Another possibility is using special programs, which facilitate work with MySQL. One of them - phpMyAdmin-2.2.0 [5] we have used to create the schedule designed before. It is the program, which should be put in a place defined through the WWW server. After having made appropriate changes in the configuration file we are able to control bases created in the MySQL using an appropriate graphic interface WWW.

3.1. Advanced searching for information in database

With no doubt, getting individually selected data is for users the most essential element. In order to do this one can prepare a menu, which makes the parallel task possible with different parameters of searching. An example of such a menu is visible in Fig. 3. This figure represents the WWW side, which is the graphic interface to the database containing information about all master theses written in the Department of Theoretical Physics in Lublin.

Fig. 3. Example of the graphic interface to the database

3.2. Administration of the database

The modification of database may be done using WWW sides. This is a very important element, because scientific servers are often used by persons who are not computer experts. Writing about service we mean: addition, subtraction and modification of information. The most complicated operation on database is the modification of the information which exist already in the database. Below I present a fragment of the code, which is responsible for such an operation:

```
1 $baza = mysql_connect("localhost","root","xxxxx");
2 mysql_select_db("dyplomy", $baza);
3
4 $table_name="dyp1";
5
6  $ed  = "SELECT * FROM $table_name ORDER BY id";
7
8  $result = @mysql_query($ed,$baza);
9
10  while ($row = mysql_fetch_array($result)) {
11    $lp = $row['lp'];
12    $datap = $row['datap'];
13    $dataz = $row['dataz'];
14    $autor = $row['autor'];
15    $temat = $row['temat'];
16    $promotor = $row['promotor'];
17    $anul=$row['anul'];
18    $idd= $row['id'];
19  $display=("<TR><TD>$lp</TD><TD>$datap</TD>                    \\
20    <TD>$dataz</TD><TD>  <AHREF=\"edytuj2dyp.php?idd=$idd\">  \\
21   <strong>$autor</strong></A></TD><TD><em>$temat</em></TD>\\
22   <TD>$promotor</TD> <TD>$anul</TD></TR>");
23
24    echo("$display");
25    }
```

   Lines 1-2 refer to the server, on which our database is situated and then to the
particular database. This code will be situated at the top of each side, because in
order to perform any operation on the database server we have to be connected
with it before. Lines 4-8 join us with the database *"dyplomy"*, and the schedule
"dyp1" of all records ordered by *"id"* number and they store the results in the
variable *$result.* In lines 10-18 one stores in created before variables the values
from particular fields from the earlier chosen schedule *"dyp1"*. Then in the lines
19-22 one writes down the obtained data to the chosen schedule, which is finally
displayed on the WWW side by a function situated in line 24. One should also
pay attention to line 20, and in particular to the element
   <A HREF="edytuj2dyp.php?idd=\$idd">, because thanks to this registration
to variable *$author* will be ascribed link, which will lead us to different side.
Moreover it will write down the data from the *"dypl"* schedule to the HTML
form. Identification of specific records is possible thanks to delivery of variable
$idd.

Now we would like to describe the side, to which we have passed the variable *$idd,* which unambiguously identifies the record in schedule *"dypl"*. Below we present the corresponding fragment of the code, this time we will skip part in which we connect to a particular schedule:

```
26  $sql="SELECT * FROM dyp1 WHERE id=$idd";
27  $result = @mysql_query($sql);
28  while($row=@mysql_fetch_array($result)) {
29
30    $idd=$row['id'];
31    $lp= $row['lp'];
32    $datap= $row['datap'];
33    $dataz= $row['dataz'];
34    $temat=$row['temat'];
35    $promotor=$row['promotor'];
36    $anul=$row['anul'];
37    $autor=$row['autor'];
38    }
39    if($edytuj): {
40
41    $ed = "UPDATE dyp1 SET lp='$lp',datap='$datap',dataz='$dataz',\\
42     temat='$temat,autor='$autor',promotor='$promotor',anul='$anul'    \\
43     WHERE id=$iddd";
44
45    $resulted= @mysql_query($ed,$baza);
46
47            }
```

In lines 26-25 one stores in the variable *$result* the information connected with the *$idd* record which was filled before. This information was obtained from the schedule *"dypl"*. In lines 30-38 one records the values taken from the schedule in newly defined variables. Then these variables are placed in suitable fields of a standard HTML form.

If one changes any value on the WWW side, which is contained in the form and if one confirms it, then the part of the program situated in lines 41-47 will be started. The new values of the data will replace the old ones, which is possible thanks to the parameter *"UPDATE",* which is situated in line 41.

Two more parts remain in the administrative menu, thanks to them we can remove or add the chosen information. The solution used there is very similar to the menu for the data edition. The difference is connected with the parameter, which is passed to the database. In the menu edition code contained the

parameter "UPDATE", in the menu remove *"DELETE"* and in the menu addition it is the parameter *"INSERT"*.

4. Summary

The example described above represents the profile of many-layered applications. We showed also one way of implementation of key elements. The use of such a type of applications is huge. They give a possibility to store various information, beginning from text data, until multimedia data, such as interactive animations of different scientific problems. Currently, the system is being projected, in which there will be information about the scientists of the Department of Theoretical Physics in Lublin. The database will also contain the sources of scientific publications as well as special numerical programs and more other information.

References

[1]   Official web page of Internet Portal "DevParadise" - http://devparadise.com
[2]   Yarger R.J., Reesw G., King T., *MySQL i mSQL*, Helion, (2001), in Polish.
[3]   Laurie B., Laurie I.P., *Apache -Przewodnik encyklopedyczny*, Helion, (2000), in Polish.
[4], [5] Official web page of PHP language - http://php.net