Sharing Acute Myocardial Infarction Databases through the Internet with MySQL and PHP: A Web-Accessible Database for Clinical Research Networks

S Carrasco[1,2], R Sanz[3], D Moratal[1], V Bodí[2], JJ Rieta[1]

[1]Biomedical Synergy, Valencia University of Technology, Spain
[2]Service of Hemodynamics, Clinical University Hospital, Valencia, Spain
[3]Service of Radiology, Quirón Hospital, Valencia, Spain

Abstract

In order to establish collaborations between different clinical research networks, a web-accessible database was set up at the Hospital Clínico Universitario of Valencia (Spain), adapted to the analysis protocol and monitoring of patients who have undergone an Acute Myocardial Infarction. Designed mainly with MySQL 5 and PHP, it offers several advantages when compared to either FileMaker or MS Access, namely a greater security in data access, low cost, compatibility and portability, and accessibility with the different operating systems and computer architectures.

The application allows different operations including, among others, making arithmetic calculations and logical operations on the records related to the magnetic resonance study variables. Currently, it contains data of about 150 patients and in the future it is expected to contain up to 500 patient data to work with.

1. Introduction

At present, within the clinical research environment where the Hemodynamics Service of the Valencia's Hospital Clínico Universitario works, we could observe that many groups are dedicated to investigate similar subjects. Although the protocols established by these groups are different one from another, they all start from the same data source: patients. The studies carried out (catheterism, ecography, magnetic resonance) offer practically the same data. Nevertheless, the way these data are used and turned into information and knowledge is different. For that reason, we consider it would be interesting to have more homogenous data that could be therefore shared among different groups (even if subsequently their statistical processing was different). So the idea involves sharing properly standardized databases which can be accessible from any place.

In this sense, the Internet appears like the ideal means of sharing knowledge among different research groups, by using classic client-server architecture model, when it comes to structuring the application design.

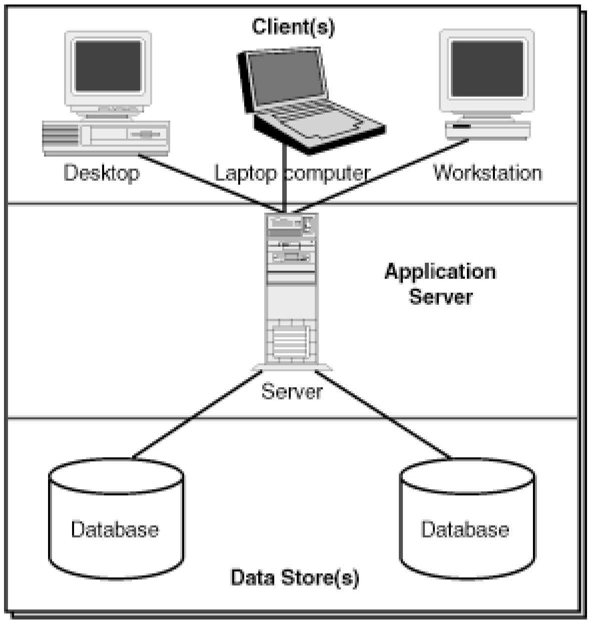


Figure 1. Client-Server architecture.

As Figure 1 shows, databases are located in a server, which is accessible by the client (user) through a computer connected to the Internet. Thinking about costs, flexibility and portability, we decided that the most advisable was to install an Apache Server [1], to use MySQL 5 [2] as a relational database manager system (RDBMS) and the PHP programming language, since it is specially suited for developing client-server applications and creating dynamic content for web sites [3]. Requests from PHP towards MySQL are carried out by using SQL (Structured Query Language). All is viable on both Windows and Unix systems. In our case, we use a Windows system, and the server is located in a workstation (HP XW6000, Xeon Processor at 2.4 GHz, 2 GB of RAM).

2. Methods

As starting point, we have a database built on FileMaker [4], where we have stored the different data related to the patients and of interest for the investigation: clinical data, data of evolution, catheterism variables, concentration of leukocytes and enzymes, genetics variables, quantification of the ST segment evolution and variables obtained through magnetic resonance imaging (delayed-enhancement, perfusion, viability). The stored data are mainly of four types: Number, Text, Date and Time. Thanks to MySQL, we can define more specific data types within each category. There are also values which were obtained from calculations among certain fields. The first option in order to create the database is a basic migration from FileMaker to MySQL, consisting of the following steps:

1. Define MySQL table name.
2. Define MySQL column names from FileMaker fields.
3. Determine maximum fieldsizes within FileMaker database.
4. Create MySQL table.
5. Transfer data via ODBC [5] from FileMaker to MySQL.

However, given the characteristics and design of the starting database, it was impossible to apply this migration process. Since FileMaker is not really a RDBMS, a database created from this software does not comply with the relational model. Therefore, its information does not represent a set of records and relationships among them. Besides, data do not have referential integrity either. Because of all this, it was necessary to perform a complete design for MySQL, almost starting from scratch.

Apart from applying a relational model to the design, the referential integrity of data is necessary as well. For that reason, it was decided to use the InnoDB [6] storage engine when defining and creating tables. Thus we are able to define foreign keys on fields, and to provide the data with referential integrity. Data must also undergo a normalization process in order either to reduce or to remove their duplicity, and have to be organized into an efficient and logical structure. This normalization process consists of applying the Normal Forms. In general, three first forms defined by Edgar F. Cood in [7] are sufficient to cover our database needs. Namely:

- *First Normal Form (1NF):* A table is in 1NF if and only if it faithfully represents a relation.

- *Second Normal Form (2NF):* The table must be in 1NF and none of the non-prime attributes of the table are functionally dependent on a part (proper subset) of a candidate key.

- *Third Normal Form (3NF):* The table must be in 2NF and every non-prime attribute of the table must be non-transitively dependent on every candidate key.

Most data either belong or are related to the patients. First of all, a table called "patients" was created where data differentiating one patient from another are stored. The primary key of this table is the patient history ("historia"), since it is a unique data for every patient. Other primary keys from the rest of tables make reference to this one. Those keys are also foreign keys. The patient table definition is as follows:

```
+-------------+-----------------------+------+-----+
| Field       | Type                  | Null | Key |
+-------------+-----------------------+------+-----+
| historia    | mediumint(8) unsigned | NO   | PRI |
| nombre      | varchar(70)           | NO   |     |
| direccion   | varchar(150)          | YES  |     |
| tlf1        | varchar(50)           | YES  |     |
| tlf2        | varchar(50)           | YES  |     |
| fecha_iam   | date                  | YES  |     |
| consultaseg | varchar(50)           | YES  |     |
+-------------+-----------------------+------+-----+
```

Figure 2. Description of the table "pacientes".

As Figure 2 shows, another obtained advantage is to define data types more strictly. For the rest of tables with information about study variables of each patient, we find an equivalent structure to the following one:

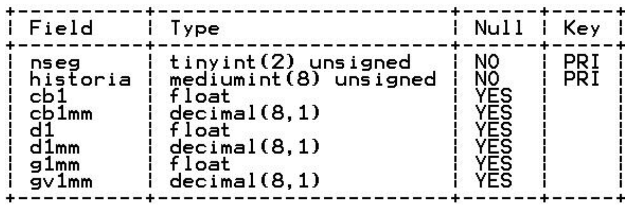
```
+----------+-----------------------+------+-----+
| Field    | Type                  | Null | Key |
+----------+-----------------------+------+-----+
| historia | mediumint(8) unsigned | NO   | PRI |
| exitush  | tinyint(1) unsigned   | YES  |     |
| relamh   | tinyint(1) unsigned   | YES  |     |
| angposth | tinyint(1) unsigned   | YES  |     |
| killip   | enum('1','2','3','4') | YES  |     |
| fa       | tinyint(1) unsigned   | YES  |     |
| tv       | tinyint(1) unsigned   | YES  |     |
| bav      | tinyint(1) unsigned   | YES  |     |
| br       | tinyint(1) unsigned   | YES  |     |
+----------+-----------------------+------+-----+
```

Figure 3. Description of the table "eventoshosp".

The field "historia" in Figure 3 is the primary key of the table "eventoshosp", and it is also a foreign key that makes reference to the primary key "historia" in table "pacientes". Thus, the referential integrity of data is ensured. However, a problem arose when sharing and making accessible the information to other groups from different hospitals. It could be possible that duplicated patient histories appeared, and this field must be a unique piece of data for each stored patient. In order to solve this problem, we had to modify the "patients" table definition, adding a new field that shows the hospital the patients belong to, besides creating a new table to store hospitals' names. These fields are defined as indices and foreign keys in order to keep referential integrity of data.

Data concerning the magnetic resonance study variables require a different processing, due to kind of information they contain. Data belong not only to a patient, but also to each of the 17 segments in which the left ventricle of the myocardium has been divided into, according to what was stated in [2]. A different definition is necessary for the tables containing these data in order to keep the referential integrity of data relationships. These changes allow implementing an efficient data export by segments and they also facilitate subsequently calculations with them. Thereby, it was necessary to create a table called "segmentos" where definitions of the 17 segments are stored.

For each table containing magnetic resonance data, the primary key will consist of two indices: the field "historia" and the number of segment ("nseg") the data belong to. Besides, every index is also a foreign key which makes reference to the table "pacientes" ("historia") and to the table "segmentos" ("nseg"), whose primary key identifies to each of the 17 segments. Figure 4 shows an example of magnetic resonance table definition:



```
+-----------+-----------------------+------+-----+
| Field     | Type                  | Null | Key |
+-----------+-----------------------+------+-----+
| nseg      | tinyint(2) unsigned   | NO   | PRI |
| historia  | mediumint(8) unsigned | NO   | PRI |
| cb1       | float                 | YES  |     |
| cb1mm     | decimal(8,1)          | YES  |     |
| d1        | float                 | YES  |     |
| d1mm      | decimal(8,1)          | YES  |     |
| g1mm      | float                 | YES  |     |
| gv1mm     | decimal(8,1)          | YES  |     |
+-----------+-----------------------+------+-----+
```

Figure 4. Description of the table "r1engrosa".

Once the first database design is finished, we start exporting data from FileMaker to MySQL, in order to help design a web interface, which is developed mainly with PHP. Content and design are separated by using XHTML and CSS, according to the W3C standards [9]. The web site is structured in several modules which allow to introduce, modify, remove, display, search and export data.

Data must be accessible for authorized clinical staff. User's permissions are granted through MySQL ("users" table) and PHP scripts. The application performs user's registration and when a user logs in, information is kept in session variables, instead of using cookies. This increases portability and compatibility, since there might be users whose browsers do not allow cookies.

In order to make calculations and logical operations, scripts have been written both in PHP and SQL in order to use MySQL mathematical functions. That means an increase in performance. For instance, these are the steps to follow when multiplying and storing the result of two records, A and B, which belong to the same table. Using only PHP code:

1. Records A and B are requested from the PHP script by a SQL statement.
2. Server runs the statement, extracts data from the database and sends it.
3. Multiplication and storage of result are requested from the script.
4. Server runs multiplication.
5. Server updates the database.

Thanks to MySQL mathematical functions, those five steps can be reduced to only two:

1. Script requests to store the multiplication of A and B: *"update table_name set C=A*B where id='value'"*.
2. Server runs the code.

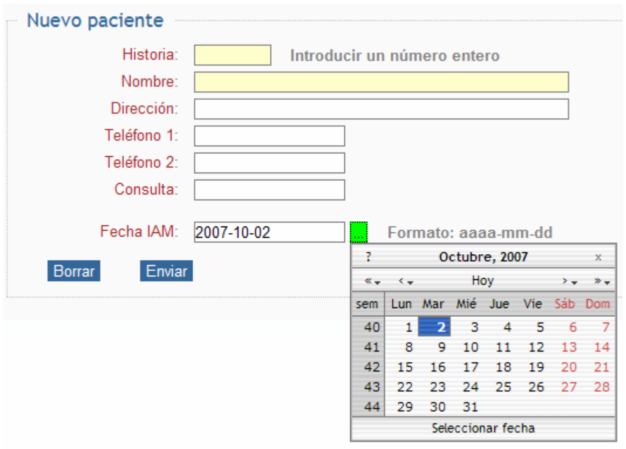
Application control, forms generation, data introduction, modification and validation functions, as well as management messages between the server and the database run under PHP scripts.

3. Results

When implementing the application and installing the necessary tools for its development and management, it was decided to use the WAMP5 software package for Windows operating system. It provides the installation of Apache server, MySQL database and phpMyAdmin from the same source. At this moment, versions of running software are: Apache 2.0.59, MySQL 5.0.27-community-nt, PHP 5.2 and phpMyAdmin 2.9.0.3.

Currently, version 1.0 is running and database contains data of 150 patients stored in approximately 70 tables. Remote connection tests were successfully performed obtaining execution times within server range. Calculations with records concerning the magnetic resonance study variables offer maximum execution time. It is worthy of mention the fact that we can establish the maximum execution time of each script, in seconds, within "php.ini" configuration file. With regard to error reporting, all errors except for notices and coding standards warnings are showed.

Even though only authorized staff can enter into the intranet where the application is located, not all authenticated users have the same permissions over data manipulation. Thus, critical tasks such as insert, update and delete are only allowed to certain users. Access to data is carried out by typing either patient history or part of patient's name. In the second case, it displays all found records similar to typed words. Export module allows exporting data both by segments and patients. User must type a file name where saving them and it can be downloaded later and imported from other software. Format file is text with tabulations, widely used by data statistical analysis software. Patients' personal data remain anonymous during this process.


Nuevo paciente

Historia:	Introducir un número entero
Nombre:	
Dirección:	
Teléfono 1:	
Teléfono 2:	
Consulta:	
Fecha IAM:	2007-10-02 Formato: aaaa-mm-dd

Borrar Enviar

Octubre, 2007

sem	Lun	Mar	Mié	Jue	Vie	Sáb	Dom
40	1	2	3	4	5	6	7
41	8	9	10	11	12	13	14
42	15	16	17	18	19	20	21
43	22	23	24	25	26	27	28
44	29	30	31				

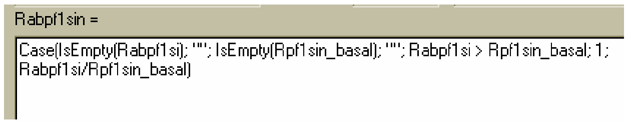
Figure 5. Introduction form data of a new patient.

4. Discussion and conclusions

This project emerges from the need of sharing knowledge and establishing collaborations among clinical research groups of different hospitals. For this purpose, tools were mainly chosen following criteria of cost and flexibility, since it is about free, portable and continously updated software.

The advantage of using MySQL 5 instead of FileMaker lies in relational model design along with referential integrity of data, none present on the last one. InnoDB storage engine ensures referential integrity of data, and although it is not as fast as MyISAM [10], execution time will not represent a problem because of stored data size. Referential integrity provided by InnoDB engine allows both deleting and updating selected records and all their dependencies. It reduces the programmer's effort as well as the number of PHP functions that would be necessary to code in order to check data integrity, to avoid duplicity and to delete by error any data.

A perfect example of efficiency is also not to have to write mathematical functions to make calculations for each existing variable in FileMaker. Considering only magnetic resonance study variables, we had to define at least seventeen mathematical functions for every variable. That was the process when adding a new variable, as Figure 6 shows.


Rabpf1sin =

```
Case(IsEmpty(Rabpf1si); ""; IsEmpty(Rpf1sin_basal); ""; Rabpf1si > Rpf1sin_basal; 1;
Rabpf1si/Rpf1sin_basal)
```

Figure 6. Definitions for every variable in FileMaker.

Now, data are not stored in variables. Since MySQL is a RDBMS, data are a set of records and relationships. PHP and SQL languages allow to simplify and to reduce code to make calculations. It is enough to program a function to work with records related to the myocardial segmentation:

```
function function_name($hist) {
        global $conex;
        for ($i=1; $i<=17; $i++) {
                PHP code & SQL statements;
        }
}
```

Everything that has been stated before is an example of flexibility, portability and efficiency when it comes to introducing improvements into the application and to extend it in the future. In this sense, communication between clinical research staff and staff dedicated to the application development is essential.

Acknowledgements

The authors would like to thank L.F. Chaustre, G. Royán, J.M Aznar, A. Frutos, B. Cabrera, C. Pérez and P. Camacho for their fruitful comments, suggestions and continuous support.

References

[1] Apache HTTP Server is a project of the Apache Software Foundation. http://httpd.apache.org

[2] MySQL 5.0 Database Server – Community Edition. http://dev.mysql.com

[3] http://php.net/

[4] FileMaker is a trademark of FileMaker, Inc., registered in the US and other countries.

[5] http://msdn2.microsoft.com/en-us/library/ms710252.aspx

[6] http://www.innodb.com

[7] *A Relational Model of Data for Large Shared Data Banks* Communications of the ACM, Vol. 13, No. 6, June 1970, pp. 377-387.

[8] *Standardized Myocardial Segmentation and Nomenclaure for Tomographic Imaging of the Heart. Circulation 2002;105:539-54*

[9] *World Wide Web Consortium. http://www.w3c.org*

[1] http://dev.mysql.com/doc/refman/5.1/en/myisam-storage-engine.html

Address for correspondence

Susana Carrasco / David Moratal
Electronics Engineering Department
Universidad Politécnica de Valencia
Camino de Vera, s/n
46022 Valencia
sucarra@teleco.upv.es / dmoratal@eln.upv.es