The Training Log

David Jacobsson and Ola Lundmark

June 9, 2004

Abstract

The Training Log is a tool that can be used by everyone from serious athletes to normally trained persons. It gives the user help to keep the training activities on the right track and to get the result he/she wants.

Outcome of the project shows that PostgreSQL and PHP is a very suitable combination for this type of web application. Together they give fast development time, however due to its loose style and limited integrated error handling it doesn't enforce good design for the inexperienced developer.

Furthermore JpGraph has shown to be an easy to use tool but in the long run there might be performance problems.

1 Introduction

Lots of people today take their training seriously. Common for many people in this group is the need to track and plan their training sessions. This is to ensure that progress is achieved in a controllable way. This project aims at giving people an easy to use web-application and can be divided into three parts.

- **Planning** - Give the user possibility to overview upcoming training sessions. Should have monthly and weekly overviews in an intuitive user interface.
- **Logging** - Easy to use forms where the user can track the results of the performed training. This will be intertwined with the planning part.
- **Statistics** - Here the user should be able to watch several different statistic summaries on performed training. Preferably the user should be able to somehow specify his/hers own queries in an easy to use way.

As a separate module to the parts mentioned above a community framework will be developed. This is done so users can share results and thoughts about their training with each other.

2 Approach

Because of our limited experience with PHP and developing web applications in general much time in the initial phase was spent on learning how to use these tools [2]. As a database back-end PostgreSQL was chosen, mostly because our prior experience with it and also because of it's good integration with PHP [3].

A quick overview of similar solutions have also been made. This was mainly to get a feel for the HCI aspects of the development and which data that would be interesting to track. One noteworthy example is running-log.com. That one is however only usable for running and lacks both a planning part and good statistics[4]. The solution for this project needed to be more general though.

2.1 Community

The community part of the web application could be solved in several different ways. In the end the choice fell on an easy to understand friend based system. This means that you can send messages to all users but private data such as training statistics can only be viewed by users that you have authorized by adding them as friends. Furthermore friends will appear in on the right of the web page for easy access (see figure 1).

2.2 Session handling

Session handling is provided by PHP with session ids in form of client side cookies or propagation through HTML GET. All data regarding the session is then saved on the server side and identified with the session id from the client. The initial authentication is performed through a database lookup with matching of user name and a md5 hashed password. No real effort has been put into making the application completely safe, but in its current state it should at least be moderately safe.

2.3 Data modeling

From the beginning the plan was to make the training log able to handle different training types dynamically. This meant that users would be able to log any type of training sessions and add new ones as needed. This approach produced two rather significant problems. First a data model that would handle this dynamic data would need to be designed. Secondly statistics would have to be generated from this dynamic data in an easy-to-use fashion.

One of the proposed solutions for the first problem that were in the end rejected was to use XML documents. XML has the advantage that new data can be added to a schema without making new documents incompatible with earlier schemas.

The second problem is that the queries on the training data would have to be generated dynamically to suit the appropriate data. To do this without making it to complex for the user is really hard and users of the system should not need any knowledge of SQL, XQuery or anything similar.

In the end it was found that there wasn't really enough drastically different types of training or meta data to be logged that would justify implementing this dynamic model, which would have taken a lot of time, instead of just implementing support for it statically. So instead of making the system dynamic by user perspective we tried to make the system as modular as possible and more dynamic from a developers point of view.

The increased modularity was designed by using inheritance where a training session is represented by a session class. Each different training type is then sub-classed from session. In the prototype only one subclass was implemented. That one was designed to handle all kinds of distance based training such as running, skiing or cycling. Adding new types to the database is relatively easy and only requires that a new subclass of session is created and this breaks no earlier code. The GUI will of course need to be updated to handle the new type and new queries must be created to generate statistics that is unique for this type. Unfortunately user input in PHP is rather inconvenient and which makes this process a bit more cumbersome.

2.4 Maps and paths

A problem that often occurs is that you don't always know how far you've run. Therefore an addition was made to the initial proposal: the path definition tool. Here the user has the possibility to upload maps to the web site and then point and click to define paths on them for distance calculation.

The path definition was implemented as a Java applet because this would give the user the best interactive feedback because no communication with the server would be necessary. At first the idea was to let the Java applet communicate directly to the database through JDBC. After some tests this solution was discarded because of the limited access rights for Java applets. In particular, applets can only access the server from which they where loaded. This restriction is bad if, as often is the case, the database server doesn't reside on the same host as the web server. The approach finally chosen was to let the HTML page "talk" with the Java applet by using Javascript. The technique is to use a script to pull the defined path out of the applet and submit it to a web page.

2.5 Statistics visualization

In the current state of the application two different types of charts have been produced: Pie charts that shows the distribution of training types and bar charts that shows, for example, distance run per week or month.

Effort have been put into trying to generate the two different kinds of charts in a flexible way. For example, it is pretty straightforward to generate the number of hours run instead of distance run in the bar chart. This have been done by doing a basic SQL query that gets data in a, for PHP, easy to read format. And making such changes does not require any modification of the PHP code.

The summation and grouping in weeks/months have been done with help of PostgreSQL's extract function. This have proven to be really useful because it really enables SQL's full potential instead of letting PHP do the job.

All charts in the prototype can also be restricted to an interval of time to allow the user to get information about a specific training period.

To get a user friendly view of the statistics generated on the web site a visualization tool was needed. JpGraph was found and because of its ease-of-use and professional look-and-feel it was chosen (see figure 1 for an example pie chart)[1].

The JpGraph tools generates a picture that is returned by a PHP script for display on the web site. This is convenient because no handling of temporary image files has to be done.

3 Results

The main result achieved in is the prototype shown in figure 1.

Since the result hasn't been the main focus of this project, but rather learning how to develop a functioning database application, this section will be kept as short as this.

4 Discussion

PHP gives you much power to write systems like the training log. Writeability is high and it's easy to get started. The problem is that if you don't know what you're doing the structure of the application can easily get out of control. This is somewhat like C++, you can do really much in many different ways, but if you don't design you application for real from scratch, you will eventually shoot yourself in the foot.

The problem with PHP and HTML, or at least the problem that we had, was that the structure of the web site and it's contents was tightly coupled. With greater experience in web design it might have been possible to avoid this. One solution we considered was to have all the content in XML format and use XSLT transformations to generate the HTML code. But since we lacked experience in this, and furthermore since it wasn't really the scope of this class, we did it the easier but more structurally awkward way.

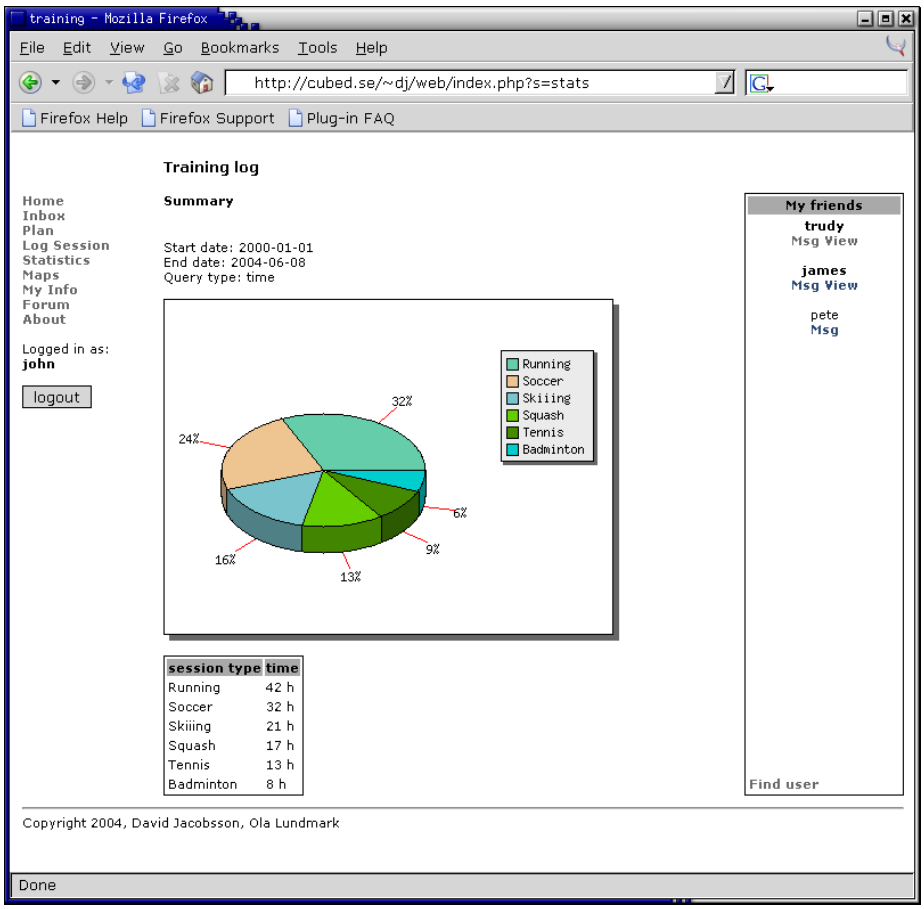

Figure 1: Statistics part of application

PHP has very good support for using a PostgreSQL database back-end. We tried to isolate all PostgreSQL specific function calls to a single script so that it would make it easy to change to, for example, MySQL or some other back end. But still far too much of the database related parts of the system was mixed together with static content and the structural parts of the web site. As mentioned before better knowledge of PHP prior to this project would certainly have helped the design of the system.

One thing that we had designed for but never got enough time to get around to implementing fully was a more ROLAP style statistics model. We designed the database schema so that training sessions could be grouped into different categories much like roll-ups in ROLAP. To group by time and date the SQL extract function could be used and no special dimension table was needed for that. Grouping by simple intervals is also easy to do in SQL and requires no additional tables. However including these features in the GUI in a usable manner was somewhat trickier and required a lot of form design which is quite a hassle in PHP.

All in all this feels pretty much as a first iteration in a project and with a second one most of these PHP related problems would certainly have been solved.

Performance results on how the web site would scale under a load of many users haven't been performed and no conclusion can be drawn. One can notice that building the monthly view in the planning section already takes some time in this application and therefore performance measurements probably should be done.

JpGraph was perfect as a tool for this small prototype. One could fear though that performance could be bad for a larger web site with more users. Caching of images

is also not an alternative when all users most of the time will be looking at different diagrams. On the other hand the charts created are good looking and generating them is really easy when the data has been gathered from the database.

5 Conclusions

The PHP and PostgreSQL combo certainly does the job of giving the developer the tools needed to create advanced web applications.

PostgreSQL's extract function has shown to be very useful for aggregation of statistics over time. This together with dimension tables for other types gives some functionality similar to ROLAP.

JpGraph is very useful for small applications for fast and easy chart generation. The results are also visually pleasing.

5.1 Future work

For us the aim of this part of the project has been to create a working prototype of the system and of course learn some valuable knowledge about developing database enabled applications. But the end of this course only means that a new part of the project has begun. Since we hope that this system will be useful, not only to us but also to other people who might benefit from this kind of system, development of it will continue for some time to come.

There are still several aspects which we wish to improve and learn more about. The statistics model we have created now, while it works very good, can be extended to support many more interesting statistics. To enable this further even more data will have to logged and the schema must be extended to handle this.

The ROLAP type dimensions is something we wish to develop further since now the only dimension which supports several layers of aggregation is time, which can be grouped either by week or by month.

Finally the community part of the system is open for an infinity of enhancements. Foremost we wish to enable users to be able to share and compare results and statistics with each other and the possibility to to co-plan their activities.

References

[1] JPGRAPH. Jpgraph – an oo graph library for php4. Webpage visited, 08 June 2004. http://www.aditus.nu/jpgraph/.

[2] PHP. Php: Hypertext preprocessor. Webpage visited, 08 June 2004. http://www.php.net/.

[3] POSTGRESQL. Postgresql. Webpage visited, 08 June 2004. http://www.postgresql.org/.

[4] RUNNING-LOG.COM. Running-log.com – your online solution for training logs. Webpage visited, 08 June 2004. http://www.running-log.com/.