

# Restructuring the Easy Learning On-line Platform

**Radu Rădescu, Radu Velțan, Raul Tudor**

Polytechnic University of Bucharest,  
Applied Electronics and Information Engineering Dept.  
1-3, Iuliu Maniu Blvd., Sector 6, ROMANIA  
E-mail: rradescu@atm.neuro.pub.ro

## Abstract

*The present paper deals with the methodology used to reinvent the Easy Learning platform, in order to facilitate the overall control of this e-learning system. The main goal was to increase the coherence in writing the code and in designing the database. Therefore, the programming errors are easy to detect and the flexibility of the platform modules is increased. The best solution was to use the Symfony architecture, for its independence on the database. The restructuring of the platform has two purposes: unification of the existing database components and standardization of the basic rules for programming.*

**Keywords:** eLearning platform, Symfony architecture, Modules design

## 1 Introduction: Implementing Problems

The need to restructure the learning process without influencing the quality of information passed on to the student is the very first problem we encounter when trying to migrate from a classic education system to *eLearning*. As long as the tutor who creates the course bears in mind the end result, which is the current level of the student and the goal level he should reach, the teaching process's quality will not suffer.

The *eLearning* term is starting to be interpreted in various ways and even if standardization is applied, it will not be able to fit the term in strict boundaries, as this term has become generic. The passing to the virtual teaching methods must be done in such a manner so that the human model will not diminish in any way its methodic and didactic skills. From this perspective, the virtualization must be done based on pragmatic, humanizes methods with applicability in real life.

This section identifies some issues that might occur and stifle the transition process and the inherent risks any change creates.

The additional work that the tutor is supposed to put into in order to modify the didactic material as well as restructuring the presentation form of that material is the first problem we encounter. Even if the manual used in the classic system contains the right information, if it is just transferred in an electronic environment the result will be far from the desired one if the material is not properly adjusted [2].

In addition, the additional work will need more advanced skills in computer science and maybe the second problem is more pressing than the first for many tutors who are

---

experts in their fields, but have limited knowledge in IT. A solution to this problem could be a prior course in computer use before the actual undertaking of an *eLearning* course, so that tasks such as manipulating web texts and e-mail correspondence become trivial. When a tutor is forced to work, only with *eLearning* tools without any background training the chance of dismissal of the entire web-based system rise dramatically.

*eLearning* is much more than a simple web page creation for a certain course; it must also involve the constant communication between the tutor and the students of the virtual class. Only in this way, the human factor can intervene in the students forming process. The simple posting of an electronic content and password-protected access is without a doubt insufficient for the implementation of an *eLearning* system.

The mentioned problems generate a third one: additional funding is needed, as expenses rise (due to overtime, or the further training of already employed staff or even creating new jobs) because the finished product is directly linked to the quality of the human resource. Besides these expenses, more are generated by the need to upgrade the infrastructure (both hardware and software) of the institution.

The funding issue is therefore a serious one, even if software is purchased (such as Blackboard, WebCT, etc.) or the platform is produced in-house (such as the Easy-Learning platform). Again, the same problem pops up: the training needed for the staff that will manage the hard and soft components of the *eLearning* system.

This section presents the risks that might occur when the mentioned problems are treated superficially, as well as the ones generated by the unsuited handling of teaching methods and course development. The major risk involved in an *eLearning* system is the students loss of interest in both this kind of teaching method and the courses included by the system. An on-line course can never substitute for a tutor's charisma and his ability to adapt to a certain situation through a subtle humor or changing the pedagogic strategies at the right time. Therefore, it is a real possibility that the rupture created by a virtual environment will cause a student to become estranged from the community. Although creating virtual communities is a priority in an *eLearning* system, nothing can really substitute for human interaction.

Although there are many on-line communication methods, both synchronous (chat rooms) and asynchronous (e-mail, forums), the loss of communication between the student and tutor, as well as between the students themselves is a real danger. Many times a student seeks in the education process a right of passage into the real world.

That is why the introduction of active mediators in forums and appointing a percent of the grade to the student's involvement in the on-line discussions are suggestions that could maintain an acceptable communication level [2].

In addition, this dependency on technology for the complete teaching process is a risk in itself. Any problems that appear in the infrastructure hide the students' access to information. That is why every precaution must be taken in order to avoid situations like the above. Although there are many risks involved, the migration to an *eLearning* system is strictly necessary. Society is evolving and the learning process seems to become never-ending, and this implicitly will lead to the creating of more-and-more on-line teaching systems. The key is to make the transition with at little risks as possible and always adapt the transition process to the society's needs. This is why illustrating and describing the problems that might occur is the first step in solving them.

## 2 The Structural Model of the Database

The structural model of the Easy-Learning platform is centered around the relational database. It stores in well-defined tables the entire structure of the software application. It is the main component of a learning platform, and a good design is imperative for the smooth functioning of such a project.

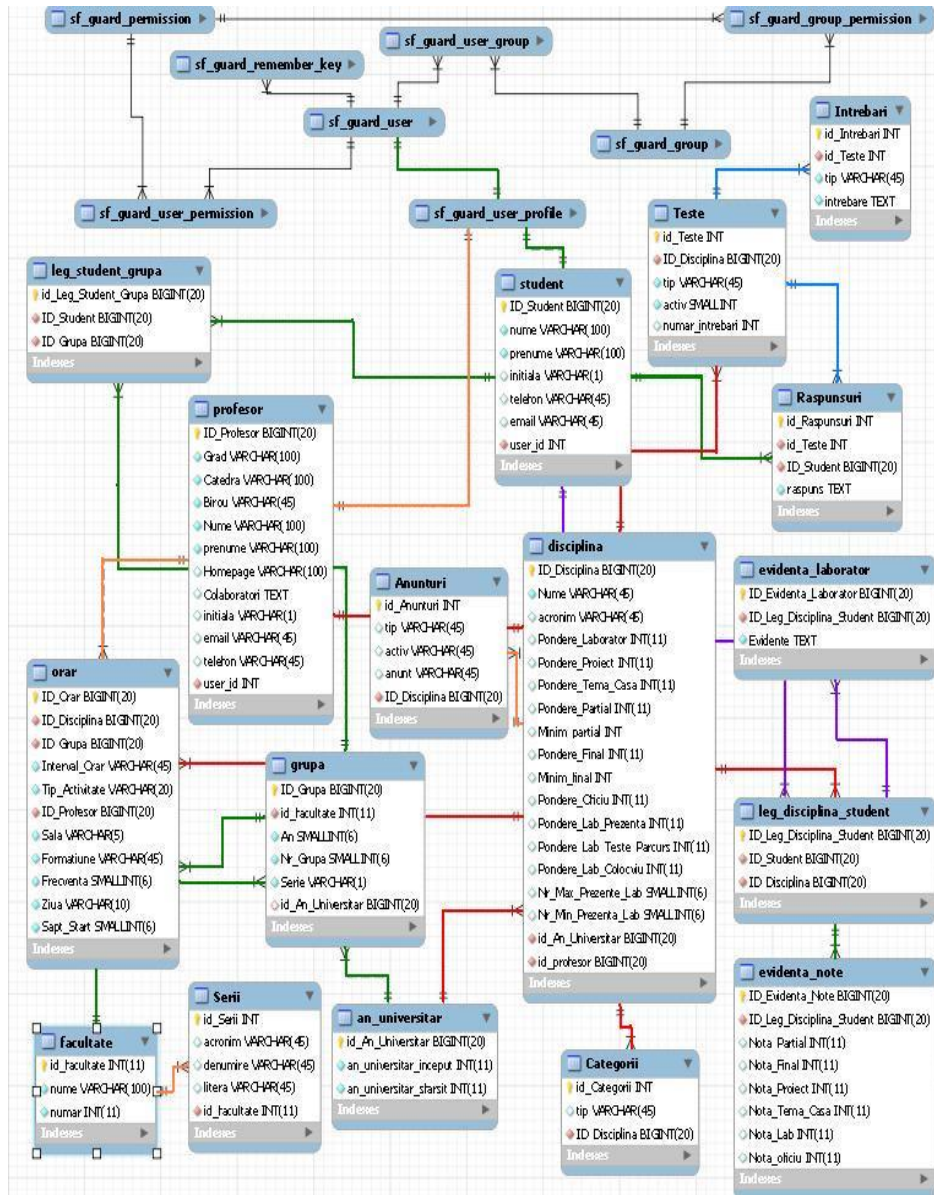


Fig. 1. The Easy-Learning Database Diagram

Figure 1 illustrates the database structure and its tables. As the database was implemented, the main factors that were taken into account were:

- Logic;
- Homogeneity;
- Simple links between tables;
- Uniform structure;
- Maintaining a buffer zone for further additions of tables and fields.

If these simple conditions are met, the project will have a solid foundation, extremely easy to handle.

At a first glance, it may look crowded and hard to follow. Actually, we tried to create very simple table relations, with as few link tables as possible. This was achieved by limiting the number of fields a table can have. This led to a higher table count, but it was a small price to pay considering the logic and control it offers.

We should note the fact that even the most complex query does not link more than three tables simultaneously, which dramatically reduces the query's execution time to a few ms, a big improvement considering the old platforms results.

### **3 The Administrator, Tutors, and Students Modules**

These three modules should be analyzed and presented together because of the close relationship that exists between them. A student or a tutor, even if they will exist separately in the „students” or „tutor” table, they both will have an associated user, through which they can access the platform.

The users module is extremely important, because this module manages all permissions and access rights in the Easy-Learning platform. It must be said that this module is a plug-in for the existing platform, and it should be explained a bit first. The entire plug-in comes with 8 separate tables in the database, and it will store every user in the database, as well as the user groups, and the permission of every user profile.

The Easy-Learning platform has three types of user groups: Administrator, Student and Tutor. Each group can access just one interface of the platform. The groups and the group's permission can be managed through the sfGuardGroup and the sfGuardPermission modules in the administrator interface. These three groups have been deemed sufficient for the current platform, and although this type of segmentation can seem stifling (as an example it can be argued that the administrator should access all the interfaces) there are strong arguments to indicate otherwise.

### **4 The Administrator Interface**

The administrator interface can be considered as the backbone of the entire application. Within this interface, users as well as their permissions are managed, as well as the school years, which are used in the new database structure as the essential separator between students and courses. As every course and student group are linked to a school year, a better management of the class book was tried, as the old platform had problems in that sense. As an example, there can now exist two student groups with the tag

„443A”, which can have very different students, or even common ones, if a student failed a year and was obliged to retake it, but each group situation is stored and interpreted differently. In addition, because a similar link exists between any course and a school year, it means we can keep a clear statistic for every year.

As we can see from the database diagram, the „course” table is in the center of the database, because of the multiple direct or indirect links it has. As data consistency it is a must in every database, an automated generator was used to create the modules which manage every table linked to the „course” and „users” tables. The current platform allows just configuring a „generator.yml” file and a completely functional module is auto-generated by the Symfony framework.

Thus, the advantages Symfony offers were used to their full potential, and the actual work was lessened, once the manipulation process of such a generator was fully understood. Although this type of generator offers more than enough advantages, there were some moments when it was not enough and certain custom actions were necessary to maintain the logical way of introducing information into the database.

Each module is secured through the pre Execute () function, which is present in every module. The Symfony framework will execute this function before anything else in each module, and in this function, the logged user’s permission rights are verified. If he does not have the right credentials, access will be denied. This check may seem redundant, but it comes to prevent a Symfony spec saying that a logged user can access any interface. Therefore, a logged student could have access to the tutor and administrator interfaces, but this preliminary check removes any doubts regarding the platform’s security.

## 5 The Tutors Interface

In this interface much of the actions undertaken by the administrator in the „course”, „notice” and „time table” modules have been kept, as it also a tutor’s job to create and manage the above modules, as the information he can have access to is limited.

As every tutor must log in order to access the interface, a major problem had to be overcome: how to display and manage only the information directly linked to the logged tutor. This issue had to be resolved, because the whole point of an authenticated interface is to limit the displayed information for the tutor, but also provide total control over that particular information.

Thus, the „course” module will only show the courses that are directly linked to the logged tutor, and the tutor will have full control over this data. The same criterion is applied to the „notice” and „time table” modules, because they are closely linked to the „course” module. This way we ensure the fact that tutor X will have full control over his information, but can’t access any of tutor Y’s information, leaving the administrator the only type of user who can access all the information, avoiding abuse and false data entries.

Besides the mentioned modules, which have also been automatically generated, just like every module in the administrator interface, the first custom build modules have appeared. These modules were harder to create, because the code had to be written from scratch.

It should be mentioned that even the custom modules have the same 4 base actions as the automatically generated ones, which are adding, editing, listing and deleting, for the following reasons:

The automatically generated code is extremely robust, but also very flexible, thus the same conduct of writing code was attempted.

The php files from the view layer could be copied from the automatically generated ones to keep coherence at a visual level throughout the application

Thus two more modules have been added, „personal data” and „documents”.

## 6 The Students Interface

The logic behind all that management in the administrator and tutor interfaces is to inform the students. Although in this particular interface the student will be able to manage just one thing, its personal data, and the real important fact is the information quantity displayed by the platform to each student.

He will have access only to the information he is interested in: which courses he is undertaking, what are his grades for each course, his colleagues, tutors and time tables for every course.

As we can see the courses are in the center of this information web, and it is represented, to be more specific, by the links created between a course and a student in the database through the administrator and tutor interface

In fact, all the work undertaken to create courses, student groups, faculties and so on was done so that the student can accumulate knowledge from only a click away, which is the end result of the Easy-Learning platform. Alongside this central goal, the tutors now have a more elegant and transparent way to manage their students and courses.

## 7 Conclusions

The Easy-Learning platform started out as a simple project, but, as the years passed, it became, though extremely useful from the student’s point of view, a very hard to control teaching instrument. Because every year somebody else appended new code to the existing one, because every programmer has a specific style, and mostly because the database had become incoherent, any bugs that had to be handled or improving an existing segment became daunting tasks.

It became clear that a ground restructuring was needed, and it had the following guidelines:

Joining all the existing databases that served the same platform into a single one, a database that could offer data cohesion and the flexibility of adding new tables as the platform grows.

Standardization of the code written and laying some ground rules as to how the code will be written and commented, so that any programmer can easily understand and debug old code, as well as writing new one in the same manner.

The Symfony platform was a logical and inspired choice. It does not depend on a certain type of database (MySQL, PosGreSQL, etc.) certain flexibility has been ensured in the case that the platform will be moved to another type of database.

Once familiarized with the framework, any programmer will be able to improve/debug old code as well as creating new one. One of the old platforms major issues, the many

code styles present in different segments, was resolved by using the Symfony framework, because it constrains the programmer to write code only in specific places, following OOP rules, so that any code will become more or less standard.

With this in mind, the actual implementing was quite easy, because of the many helpers Symfony provides (sfGuard plug-in, module generators and page generators).

The issues we encountered were technical ones, where the module generators (which are the foundation of the administrator interface) were not complex enough on their own. Still because of Symfony's flexibility, any action within the generated modules could be override in any way a programmer desires, so that the module meets the specifications.

This new platform was not built to add new facilities, but to transpose the old ones in a new shape, a shape that is much more manageable and maintainable. Using the Symfony framework, a robust and heavily tested framework proved to be a very inspired choice, which can now permit the new platform to grow in an organized manner, as it now has a flexible database to rely on. We can safely assume that the reorganization of the Easy-Learning platform was a complete success.

#### REFERENCES

1. Rădescu R., Urse C. (2007): Graphic Tools in the Easy-Learning Platform. In *The Symposium TEPE „Educational Technologies on Electronic Platforms in Engineering High Education”*, Technical University of Civil Engineering of Bucharest, Bucharest. ISSN 1843-2263.
2. Nagy, A. (2005): E-Content: Technologies and Perspectives for the European Market, in *The Impact of E-Learning*, Berlin, 79-96.
3. Băăth, J. A. (1982): Distance Students' Learning – Empirical Findings and Theoretical Deliberations, Stockholm, 30-32.
4. Scott W. A. (2000): Mapping Objects to Relational Databases: O/R Mapping in Detail, Practice Leader, Agile Development, IBM, Software Group.
5. Boodhoo J. P. (2006): Design Patterns: Model View Presenter, Microsoft.
6. Rădescu R., Urse C. (2007): Advanced Testing Methods in the Easy-Learning Platform. In *The 8-th European Conference E-COMM-LINE, SIV-26e/1...6*, Bucharest.
7. Rădescu R., Bojin M. (2006): Function generators in the Easy-Learning Platform. In *The National Conference of Virtual Education “Virtual Learning – Virtual Reality”*, Educational Software & Management, 4th Edition, University of Bucharest, Mathematics and Informatics Faculty, Bucharest, 115-120.
8. Rădescu R., Iovan R. (2005): Generating the class register in the Easy-Learning platform. In *The National Conference of Virtual Education “Virtual Learning – Virtual Reality”*, Educational Software & Management, 3rd Edition, University of Bucharest, Mathematics and Informatics Faculty, Bucharest, 213-220.
9. Rădescu R., Iovan R. (2005): Creating and using tests in the Easy-Learning platform. In *The National Conference of Virtual Education “Virtual Learning – Virtual Reality”*, Educational Software & Management, 3rd Edition, University of Bucharest, Mathematics and Informatics Faculty, Bucharest, 229-235.
10. Rădescu R.: E-learning: concepts, implementation and applications, *IT&C Market Watch, Fin Watch*, 50 (no. 30/2004, co-author Lăcraru C.), 61 (no 31/2004), 50 (no. 33/2004), Bucharest.
11. Rădescu R., Iovan R. (2004): Improvements to the Easy-Learning E-learning Platform. In *The 5-th European Conference E-COMM-LINE*, Bucharest, 275-278.
12. Rădescu R., Iovan R. (2005): New Facilities of the Easy-Learning Platform, in *Proceedings of the Symposium “Educational Technologies on Electronic Platforms in Engineering Higher Education” (TEPE 2005)*, Technical University of Civil Engineering of Bucharest, 27-28 May 2005, Bucharest, 219-226.

13. Rădescu R., Mărescu R. (2005): External Use of the Easy-Learning Platform: a Web-Based Application. In *Proceedings of the Symposium "Educational Technologies on Electronic Platforms in Engineering Higher Education"* (TEPE 2005), Technical University of Civil Engineering of Bucharest, 27-28 May 2005, Bucharest, 227-234.
14. Rădescu R. (2008): Class register optimization in the Easy-Learning platform. In *The National Conference of Virtual Education "Virtual Learning – Virtual Reality"*, Modern methods in Education and Research, 6th Edition, University of Bucharest and „Ovidius” University of Constanța, Oct. 31 – Nov. 2<sup>nd</sup>, Bucharest, B-7-55/1...4.
15. Rădescu R. (2008): Multiple tests in the Easy-Learning platform. In *The National Conference of Virtual Education "Virtual Learning – Virtual Reality"*, Modern methods in Education and Research, 6th Edition, University of Bucharest and „Ovidius” University of Constanța, Oct. 31 – Nov. 2<sup>nd</sup>, Bucharest, B-6-54/1...4.
16. Rădescu R. (2007): Test user interface in the Easy-Learning platform, *The National Conference of Virtual Education "Virtual Learning – Virtual Reality"*, Modern methods in Education and Research, 5th Edition, University of Bucharest and „Ovidius” University of Constanța, Oct. 26-28, 2007, Bucharest, 85-92.
17. Rădescu R. (2007): Test management interface in the Easy-Learning platform, *The National Conference of Virtual Education "Virtual Learning – Virtual Reality"*, Modern methods in Education and Research, 5<sup>th</sup> Edition, University of Bucharest and „Ovidius” University of Constanța, Oct. 26-28, 2007, Bucharest, 75-84.
18. [http://www.symfony-project.org/book/1\\_0/](http://www.symfony-project.org/book/1_0/)
19. <http://www.zend.com/zend/zend-engine-summary.php>
20. [http://forge.mysql.com/wiki/MySQL\\_Internals](http://forge.mysql.com/wiki/MySQL_Internals)