Evolution of the Design and Implementation of Tutor: A Web-Based Educational System for University Courses

Roberto F. Arroyo, Miguel J. Hornos, and Rosana Montes

Dept. de Lenguajes y Sistemas Informáticos, E.T.S.I. Informática y de Telecomunicación, Universidad de Granada, C/ Periodista Saucedo Aranda, s/n, 18071 Granada, Spain
{robfram,mhornos,rosana}@ugr.es

Abstract. We have designed a web platform to support the teaching and learning of certain subjects taught in the University of Granada. This platform facilitates students and teachers diverse educational tasks and provides an effective management of academic data. Originally, it was intended for a reduced array of subjects, but as a result of its increasing success and use in a greater number of subjects, we have been encouraged to extend it. However, the approach used in the early design was a classical life cycle and did not take into account either future needs or its hypermedia system nature, which is even more important and requires a special processing, which is different from the one required for conventional applications. From our experience, we have learned and have taken into account the detected lacks to make the platform progress, and we have provided a new design thus changing completely its philosophy, structure and interface. We will hereby describe the main characteristics of both designs (the first and the new one) in order to highlight the evolution of the system.

1 Introduction

New technologies have extended to the global environments of our society, facilitating many activities and admitting tasks that were unlikely years ago. In education, the teaching staffs have recognized the need for effective tools to manage information resources for their students' progress and for their subjects. The European Space of Higher Education [1,2] promotes the employment of new technologies in order to increase the cooperation among universities from its member countries and improve the educational process quality in them. In recent years, and in the context of this new educational framework, the development of electronic communication channels has allowed higher levels of cooperation within the educational community. Nowadays, computers are everywhere and university students are able to communicate no matter where they are. ICT (Information and Communication Technologies) as the World Wide Web [3] have become one of the main mechanisms for remote interaction, and have reshaped both society and universities all over the world. Universities must capitalize on the web for teaching and learning, and the increasing use of web-based systems is a progressive manner to be on the way.

R. Moreno-Díaz et al. (Eds.): EUROCAST 2007, LNCS 4739, pp. 352–359, 2007.

According to these needs, the University of Granada has created several plans of educational quality [4,5]. Within them, we are developing a teaching innovation project with the purpose of building a web-based platform, called Tutor (http://tutor.ugr.es), which provides a toolkit to support the educational activities of both teaching staff and students.

This web-based educational system, which is particularly intended for university courses, must provide an adequate pedagogical approach in order to increase its usefulness and achieve the following main proposed objectives:

- To provide additional information focused on academic contents, such as subject syllabus, didactic material, glossaries, learning activities, etc.
- To establish new communication and interaction channels among the different academic members.
- A better management of all registered users provide a secure authentication mechanism to preserve the users' identity and privacy of their data.

The system we developed in the beginning was adequate for a limited number of users and subjects, but due to its design, its adaptation to such a big growth (in users, contents and specially in functionality) requires a disproportionate effort. This mainly occurs because the hypermedia nature was not taken into account; thereby, the information storage was not separated from navigation aspects; a formal user modelling was not performed and mechanisms to carry out a satisfactory adaptation were not defined. For all these reasons, we have decided to develop a new version of Tutor, which, as well as improving the user interface and navigation system, admits adaptiveness and adaptability.

The rest of the paper is organized as follows: Section 2 will present the context and design structure regarding the first implementation of the platform. Section 3 will show the architecture and design principles followed in the development of a new version of the system. Finally, in the last section the conclusions will be outlined.

2 Initial Design

The platform initially designed follows a role-driven approach, using a role-based model. Firstly, the system roles are distinguished and the different parts that made up the whole system are developed. Next, we establish relations between the roles and parts using their operations, which will be implemented on a later developing stage. For operation modelling we use a user-driven design, which provides a higher abstraction level for system task description and separates the concepts of users and roles.

Using a role-driven approach allows us to get different functionalities according to the role played, which is reflected on the user interface. The system distinguishes between four different user profiles (see Figure 1): administrator, teacher, student and unauthenticated user. When users log in the system, it automatically recognizes the roles they play. Once access has been granted, the system displays a menu with the sections and actions allowed in conformity with the role the user is playing.

The system is based on a client/server [6] approach with dynamic content generation. It uses Apache [7] as a web server of dynamic pages, which is

accompanied by MySQL [8], a relational data base used for data storage. A collection of PHP [9] scripts implements operations and generates the HTML interface. This part is server side, both for storing and executing. We use a Sun server to support many simultaneous connections. As a client, the user can use any HTML[10]/CSS[11] compatible web browser to access the platform from any place with an Internet connection. In brief, data manipulation is not separated from the own functionality of the application, so that queries and modifications to the database contents are carried out together with the code processing, thus adding complexity to the code updating and data maintenance.

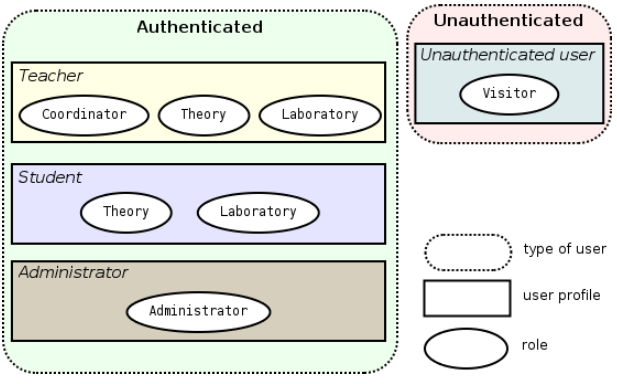

Fig. 1. Structure of user profiles and roles

The main problems presented in this implementation are the lack of some important features. Among the latter we can highlight the following ones:

- The whole system is short of friendly navigability. A series of menu items and information nodes are not easy to find, and many clicks are needed to access some of them.
- It has not a clear look-and-feel interface. The appearance of the system is neither very elegant nor intuitive.
- The implementation is not modularised, since the same or similar functionality is reimplemented in several fragments of code. This increases the number of programming errors and makes its maintenance difficult. At the outset, every role had its own functions, but several roles had the same functionality. Since every role has its own implementation, then, for the same functionality, there are different implementations in the system. Moreover, the latest implementations consider all the roles sharing the same implementation: internally, the function determines which portion of specific code must be executed according to the role played by the user. All this leads to a non-homogeneous implementation.
- The system does not support adaptability or customisation. Different users cannot have different interfaces.
- It is quite difficult to extend the system with new functionalities without breaking the system. It has not been designed to be extended easily.

- The maintenance of the system is hard. Due to some of the above mentioned items, fixing problems in the whole platform is a hard work.
- It does not follow the XHTML standard [12], using only the HTML document definition.

3 New Design of the Platform

The motivations for a new design from scratch are to solve the lacks of the earlier model while maintaining the same goals of the previous version of the platform. Therefore, we are looking for a 24/7 web-based system to support educational tasks and academic data management that provides additional student-student, teacher-student and teacher-teacher communication channels. Basically, the differences were conceptual, so that we do not want to ignore its hypermedia nature, developing it under the object oriented paradigm, separating contents from presentation and using an implicit role assignment method. With all this, we pretend to avoid and solve the most important problems of the previous version, namely:

- A greater number of users requires different and new functionalities.
- Some users get lost trying to use some of its current applications.
- The maintenance is expensive due to a redundant code.
- The navigability should be more intuitive and direct (with less mouse clicks).
- The system should be scalable and future-prone.
- The new XHTML/CSS technology should be used.
- It would be interesting to provide a multi-language platform, since the mother tongue of many users (e.g. foreign students) is not Spanish.

Tackling these main goals and features, we have designed the new model for the platform implementation.

3.1 Design Philosophy

The previous design was oriented to the role (or user profile), directing both the way in which the data were stored and the form of presentation and navigation to the role played by the user navigating in the system, independently from the part of the system being visited. The new design completely changes this orientation, proposing an object oriented design (data and operations), where what is visited, and not only who visits it, is also important. The first-level objects we have considered include the system users, subjects, degree courses, groups, faculties and departments, as well as notices and internal messages. All these objects constitute the system navigability core, and are the main information structure of the system; hence, their design deserves a major consideration. The new design is divided into two layers (see Figure 2): data storage and adaptation to each user. The connection between them is the user model, which stores, maintains and updates the user's main features.

As we have already mentioned, one of the main objectives of the new design is to separate contents from presentation. This admits multiple presentations (taking into account look-and-feel, language, restrictions, etc.) associated with a given piece of contents. In this design, the data is stored in a relational database, and the system queries the database to display the contents. Therefore, the data storage layer converts

relational data into the object oriented structure to be shown. Later, the selected contents will be published according to navigability restrictions, user capabilities and preferences, and style templates.

Another important objective is modularity. Common functionalities must be implemented only once and be shared among modules. Additionally, all the database queries are encapsulated in a separate module, outside any other module. Finally, to make up the final appearance of the information, we use XHTML templates with pattern replacing.

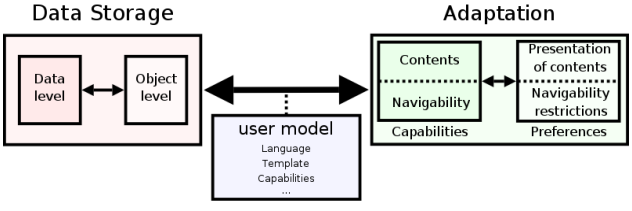

Fig. 2. New design of the system: two independent layers linked by the user model

3.2 Structure

The data storage layer contains the information. This layer is composed of two levels: the data level and the object level. The first one deals with the relational database, whilst the second is responsible for converting object-oriented queries into relational ones and relational data into object oriented structures.

The data presentation related to the personal and hypermedia criteria is carried out in the adaptation layer, which is made up of two different levels. The level of capabilities is responsible for determining which contents are displayed in accordance with the navigability patterns and with the relation with the user himself, e.g. a teacher has access to their students' e-cards, but not to those belonging to students that are not enrolled in any of his/her subjects. The level of preferences modifies the presentation and filters the contents in accordance with the user preferences and navigability restrictions.

Figure 3 shows the internal structure of both layers, and an example of the conversion of relational data into an object oriented structure carried out by the object level.

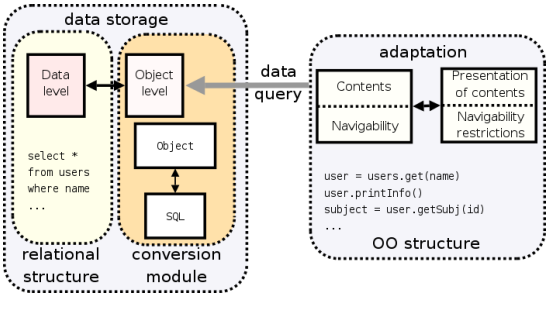

Fig. 3. Structure of the new model

3.3 Template System

The different presentations of the contents are prepared using a template system. We have defined the following four different template levels:

- *System template*, which is the first template the system uses. It shapes the main appearance and structure of the whole system, and determines which parts must be filled with next templates.
- *User model template*, which admits changing parts of the system template according to the data stores in the user model. For example, accessibility issues can be attended in this template.
- *Capability template*, which adapts the general appearance according to the user capabilities. This allows customizing the user interface and navigability of teachers, students, administrators, or the appearance of a determined object in relation to a specific user.

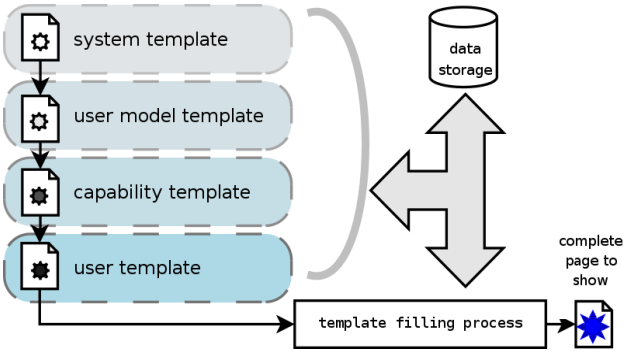

Fig. 4. Template processing: The final web page is generated by making up four different templates and accessing the database

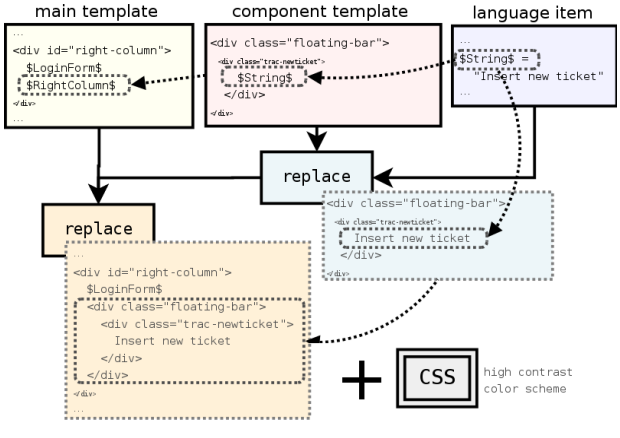

Fig. 5. Replacement of key values to convert templates into the final document

– *User template*, which lets the users customize their user interface. In such a way, every system user is able to choose certain aspects in order to customize the appearance of the contents presented by the system.

Figure 4 shows our sequential processing of the templates, from the system template to the final web page to be shown.

The final contents of the web page to be displayed are obtained by successively replacing key values (included in the different templates) with the real data, which could be from a simple string to another group of key values. The final document is obtained after all the key values are replaced. Figure 5 shows an example of this replacement.

4 Conclusions

We have presented how the design and implementation of a web-based educational platform has evolved from its initial version to the new one, in such a way that the latter constitutes an example of applying quality design principles to a hypermedia system. The evaluation of the first version of the platform showed that it lacked the essential basis to be easily flexible and adaptable to users, being necessary to spend a considerable effort to add new functionalities, as well as a more intuitive navigation system. We have also learned that the followed role-orientation, without taking into account the visited information, did not contribute to improve the system navigation.

The main problems detected in the previous version have been solved with the new design, while its positive aspects have been kept. The perspective change from role-orientation to object-orientation allows a more intuitive and controlled navigation, with a better structured presentation of the information. This admits an easier expansion and upgrading of the platform to new requirements and needs, in such a way that it is at the same time more scalable, and does not affect negatively the provided functionality. The active role of the system to make decisions has also been increased, by replacing the manual selection of the user profile with an automatic query to determine the user's capabilities while the user visits any page of the system.

The new design establishes two independent but interrelated layers: the data storage layer and the adaptation layer. The link between both layers is the user model. The data storage layer is structured in two levels: the data level and the object level. The former maintains a relational representation of the information and communicates with the DBMS, while the latter builds extended objects in accordance with the existing relations among data and communicates with the adaptation layer. The adaptation layer performs an adaptive process in two levels, deciding respectively what to show (i.e. the contents and their navigability) and how to show it (i.e. the presentation of these contents taking into account the navigability restrictions, which can hide part of the selected contents to this user). The first level depends on the user's capabilities, while the second one depends on the preferences established by each user.

The implementation of the new version of the platform is in an advanced state, but not finished, so we are currently working in order to finish its development, with the aim of replacing the previous version. We also plan to increase the functionalities of

the new system, and, at the users' request, extending the available tools with new options and operations and adding new tools to the ones already included in the platform.

Acknowledgments

This work is financed by an Educational Innovation Project from the University of Granada (code 05-03-46).

References

1. Ministers responsible for Higher Education: Realising the European Higher Education Area. Berlin, Germany: Official Communication (2003)
2. Treasury of Education, Culture and Sport: The integration of the Spanish university system in the European Space of Higher Education. White paper (in Spanish) (2003)
3. Berners-Lee, T.: Weaving the Web: The original design and ultimate destiny of the World Wide Web by its inventor. Harper Collins, New York (1999)
4. Vice-chancellor of Planning, Quality and Educational Evaluation, Plan of Educational Quality, /2004. University of Granada, Granada 2001 (in Spanish) (2001)
5. Vice-chancellor of Planning, Quality and Educational Evaluation: Plan of Educational Quality, /2008. University of Granada, Granada 2005 (in Spanish) (2005)
6. Edelstein, H.: Unraveling Client/Server Architecture. DBMS 7(5), 34–41 (1994)
7. The Apache Software Foundation, online at, http://www.apache.org/
8. MySQL AB, online at, http://www.mysql.com/
9. Welling, L., Thomson, L.: PHP and MySQL Web Development. Book & CD edn, Sams, Crawfordsville (2001)
10. W3C: HTML, online at, http://www.w3.org/html/
11. W3C: CSS, online at, http://www.w3.org/Style/CSS/
12. W3C: XHTML2 Working Group Home Page, online at, http://www.w3c.org/MarkUp/