CLASSROOM AND RESOURCE AVAILABILITY MANAGEMENT PORTAL - CRAMP: PROPOSAL, PROCESS AND FEATURES

João M. Nascimento [1]

[1] Instituto Politécnico de Santarém, Escola Superior de Gestão e Tecnologia de Santarém, Portugal, joao.nascimento@esg.ipsantarem.pt.

Abstract

Wherever there are scarce resources, decision support systems are welcome to overcome the problems of managing the allocation and availability of such resources. This paper presents a solution to aid operational level actors involved in the management of classrooms and pedagogical equipment within a scholar environment. This paper present an IT solution that aims to solve problems related to availability of resources. It will also describe the requirements elicitation to the building of the system that will help in the decision making process related to allocate a resource to a particular need, always trying to use it as efficiently as possible. As we will discuss in the paper, information storage, integration, and communication are the most important aspects to consider. There is enough evidence that the operational level errors related to this point will be reduced.

Keywords: Software Engineering, Web Portals, Decision Support System, System Analysis and Design.

Resumo

Onde quer que exista escassez de recursos, os sistemas de apoio à decisão são sempre úteis, no sentido de viabilizarem uma alocação adequada e uma gestão da disponibilidade desses recursos. Este artigo apresenta uma solução para suprir uma dificuldade operacional, com que se confrontam os agentes envolvidos no processo de gestão de infraestruturas, como salas, e outros recursos pedagógicos, no contexto escolar. A solução tecnológica apresentada tem por objetivo resolver problemas relacionados com disponibilidade de recursos. Aqui será apresentada, desde o momento da caracterização do problema a resolver até à construção e implementação do produto que ajudará no processo de tomada de decisão de alocação de recursos a

um fim específico, sempre com o intuito de tornar a utilização destes recursos o mais eficiente possível. Como veremos ao longo do artigo, o armazenamento de dados, a integração e a comunicação são os aspetos mais importantes a considerar. Existem claras evidências de que se podem retirar proveitos consideráveis, pela utilização do sistema descrito.

Palavras-chave: Engenharia de Software, Portais Web, Sistemas de Apoio a Decisão, Análise e Conceção de Sistemas.

1 INTRODUCTION

This research was conducted following the Design Science for Information Systems methodology (Hevner, 2004). Thus, this paper describes the construction of an IT Artifact (Orlikowski, 2001).

The CRAMP – *Classroom and Resource Availability Management Portal* is a technological solution, built for the *Web* environment, aiming to help operational level personnel, namely teachers and administrative staff, managing the daily usage of some resources at school – namely the pedagogical resources. Although this solution is designed to support the managing of resources by school staff, and training professionals in general, we believe that this is a global problem, not only to the intended organization, but to others. As we will see, however, there are some minor details that make the solution specific to the intended organization.

The solution presented here was built with the aim of facilitate the daily job of controlling the usage of educational resources at school – classrooms and equipment available. It will help almost all levels of decision making around the pedagogical usage. The main users will be the supervisors and the teachers.

The portal – CRAMP – was developed through the application of a process composed by two complementary phases. Firstly, the analysis and design, where the steps were performed following the Ripple methodology (see O'Docherty, 2005). This methodology is based on the Unified Modeling Language (UML), introduced by Jacobson et al. (1998). Ripple suggests a sequence to align the tasks included in the

original UML specification (see OMG, 2003), so the work is done in the most convenient way. Doing this arrangement, the isolated processes became steps of the methodology. A set of new deliverables was developed to enhance the outcome of the analysis and conception of the system being constructed.

In the second stage of the development process, the system was built according to the solution designed at the first stage. The appropriate tools and techniques were chosen and applied.

This paper is structured as follows. Section 2 describes the context where the product was born, the problem being addressed and the objective of the work. Section 3 describes, in more detail, the path followed to achieve the goals, i.e. the outcomes of the methodology's major steps. Section 4 briefly presents how the solution is structured, in terms of conceptual, logical and physical levels, presenting how the requirements are met. Finally, section 5 presents some concluding remarks, the discussion about the achieved results, and some possibilities for future work.

2 FOUNDATIONS OF CRAMP

During the operational activities at school, with the rising number of classes happening and the constant (or even reduced, due to equipment breakdown) number of resources, it's frequent to observe problems related with the usage of those resources. The need for an information system to help controlling the availability of educational facilities (namely classrooms with specific resources) was obvious.

The process of room reservation is aiming to host the various activities has been done manually, registered on personal agendas or on wall calendars. There are several problems arising from these methods. Essentially, it is very difficult, to anyone who needs an educational facility, to know if it is available for an activity, on a specific date and time.

Initially, trying to avoid this issue, the room occupation schedule was posted near the door of each room. However, other issues aroused. The constant updates to the timetables of the courses, makes the information rapidly outdated, and this generates

costs related to the recurrent printing of updated timetables. Otherwise, when standing at the room's door, looking at the schedule posted there, it is very difficult to know exactly if the room is free in the next minutes. It could be available at that moment, and there's nothing foreseen to happen there, at least on the schedule on the wall. However, someone may have booked a reservation involving that room (few moments before, with the responsible for that room), starting in the next 10 minutes and taking, for example, one hour, and there no other way to know it besides asking the responsible for that room. This type of information is registered only on a personal agenda at the administrative services of the school.

Another problem comes from the office hours. If someone needs this information by 6 pm, the services are no longer available to give support. The same applies if someone wants to make a reservation early in the morning.

If a professor wants to know the availability of a specific room two months from now, it is impossible without talking to the person responsible for this task. Even, if this professor wants to know which room is, or will be, available considering the existence of a specific resource (video projector or computers or a software application), it's not easy without the help of the responsible for the rooms.

Finally, the distributed responsibility on room occupation, makes the job even harder. Each room have a responsible, and it is hard to coordinate the availability of timetable committee, and each room's responsible.

Until the initiation of this project, many were the situations of conflict because room occupation involves different educational agents. Moreover, the process must consider, not only the reservations from inside but also those made from the outside of the school – enterprise that rent classroom at the school with the purpose of training of its internals. However, there won't be access from the outside users. This is an internal system. The reservations from outside must be made via inside staff.

So, the necessity for this system is supported in three main aspects:

1) Integration – the system must centralize all the information about reservations;

2) Availability – the system must be available always (24 x 7 x 365);

3) Security – the system must control who is granted to do the different functions, and register who made the operations.

3 METHODOLOGICAL PROCEDURES

This project aims to develop a software to serve an information system. The method was applied in two separated and interconnected moments: (1) selection of the adequate methodology; and (2) its application. Here there is the initial basic description and on the next section the details about each step of the method appliance will be presented.

3.1 Selection of the Software Development Methodology

The choice of the methodology for developing the system was made based on some references found in the literature. Before the decision was made, some criteria were set. The two initially established premises for the selection of the development methodology was that it should be consistent with the object-oriented approach, and it must be adaptable. Based on these assumptions, some of the methodologies were identified: RUP - Rational Unified Process (Kroll & Kruchten, 2003), GRAPLE (Schmuller, 2004) and Ripple (O'Docherty, 2005). All these identified methodologies are based on the UML, but Ripple was adopted because of its characteristics – mainly due to the customization possibilities since all other are consistent with the object-oriented approach.

The choice of the methodology and its customization were performed having in mind the characteristics of this project. Only those deliverables considered relevant to the work were executed in a way that enables the construction of a functional portal, considering the team of developer available.

The project elements produced were organized, according to the author of the methodology, in five groups: **Genesis** (informal requirements); **Requirements** (Actor List, Use Case List, Use Case Diagram, Use Case Description, Activity Diagram and User

Interface Design); **Conception** (Class Diagram and Database Schema); **Construction** (Source Code); and **Deployment** (Components).

3.2 Selection of the Software Development Methodology

The deliverables resulting from the first group is a description of the informal requirements. This was accomplished based on the interviews made to the actual agents of the information system. As referenced before, the main requirements of the system are derived from the following well accepted principles:

1) Integration – the system must centralize all the information about reservations;

2) Availability – the system must be available at any time and any place using an internet connection;

3) Security – the system must control who is granted to do the different operations, and register who made the operations.

The system's core data are the timetables of the courses at the school. The principal occupation of the rooms is due to classes of the regular courses, and it is stated on the timetables.

The other reservations are very specific and are made by professors for, eventually, meeting with students, making an exam or a written test, a project activity, and whatever may need a space with a specific set of resources.

The Timetables Committee produced the timetables and published them before the start of each semester. After the beginning of the semester, updates to the timetables of the courses may occur (and this often happens), and there'll be another publication, whenever it is necessary.

The timetables are generated using a **Microsoft Access** application, and this data must be considered for this system.

Also, the start and the finish dates of the effect of timetable (the period when timetables are active) will change during the semester (every time a new version of

timetables is published there is the need of importing it to this system with the additional data about the effective dates). A unique user - timetable supervisor - will be responsible for the update of the system and, eventually, the setting up of the semester. During the data importing, the user will tell which date to consider as adequate interval.

When the timetable information is imported, conflict/overlap reservations are detected. If reservation conflicts with new timetable, system sends an email to professor (informative message to the one who did the reservation) and to the responsible of the room involved (with a link to the page where the conflict may be solved). It is expected that the responsible solves the conflicts identified, with the agreement of the user(s) involved (for example, assigning a different room, or simply deleting the reservation, leading to the need of a new reservation to be made by the professor or owner of the affected/deleted reservation).

The system does not permit doing reservations involving past dates; only future dates will be allowed. Users can insert reservation to a date outside of the academic year, but always in future dates.

The user (professor or room responsible) may register a new reservation of a specific room for a single moment in time. There'll be a responsible for each room. Every time a user tries to make a reservation on the system, the responsible will be always asked to approve it. The system will assure the necessary communication between those interested in the reservations and the responsible for the rooms being reserved. The users will always have access to the information about all reservations involving the pertained room, even to those not approved yet by the responsible. At least they get to know about the intentions.

The email address of the users must be registered on the system. Every message sent by the system will use the actual email address of users.

All operations that force the system to fire emails use predefined messages configured on the system. The text of this email messages is standard and specific to the type of operation that generates it. Some of the messages may include links to some of the

functionalities of the system (confirmation of reservation, and so on) to help users get directly (after authentication) where the intervention is needed.

When a user adds new reservation, the system sends an email to the responsible (with the name of the user that made the reservation and a link to the reservation record on the system). When responsible click on the link, the occupation of that room is presented.

If this new reservation gets in conflict with other previously registered reservations, it will appear on the "list of reservation conflicts". Otherwise, it is added to "pending approval reservations list". Until the problem is solved, the system continues to give a warning signal. This function is the main reason to consider this a decision support system: the user is advised to choose another room after reservation collision is detected. Another room with the same characteristics is suggested.

As one of the main features of the system is to be available anytime, anyplace, the Internet was the selected environment to develop and explore it. Initially, the possibility of integration with the Academic System – SIGARRA – was talked, but it was not be implemented, because, as it is a third party product and no commitment was possible from the enterprise responsible for the maintenance of the system, the development will be considering an independent environment. The advantage of SIGARRA integration will be the authentication process. The actual users will not have to authenticate separately to use the CRAMP. Since this integration aspect is not to be considered, there will be the need to develop authentication process too.

The development will be done using PHP and MySQL, so these are the tools that best matches the characteristics of the system and the skills of the developers.

For security purposes, multiple types of users will be considered, with multiple levels of functionality. So, after authentication, each user has a specific set of functionality at their disposal, as will be presented in the documentation of the project ahead.

Every page on the system - the components of this product - will have the responsibility to check if the actual user have the adequate permission to access it, as soon as the user loads the page. This will avoid security problems.

4 THE CRAMP

The following subsections will be devoted to the description of the resulting system, by presenting the steps from the conceptual and technological perspective, and by highlighting some of its features through the presentation of graphical user interfaces. Although on a minimum set of features, the portal may be classified as a decision-support system. Its characteristics and nature, whose structure includes a mechanism of suggestions to different users on different usage of the Portal, turns this tool into a powerful help to avoid overlap and the operational problems that usually happens.

4.1 Conceptual Basis

As expected in the Genesis phase of the RIPPLE methodology, informal requirements were gathered based on meeting. A document – report – was produced and distributed among the participants. No changes were suggested, and the next phase begin.

The Requirements phase started with the identifications of actors, and the functional requirements were detailed and presented, using a proper set of UML diagrams. The diagrams were the base for the requirements discussion (review and consolidate), with the initial participants on its elicitation. A solution was proposed, through the adequate UML diagrams, also considering the technical requirements (i.e., security, performance, and usability). The construction of the portal began when all the requirements were deemed to be satisfied by the proposed model.

The requirements elicited initially allowed to identify three principal users: Room Responsible, Professor, Timetable Committee. Then, at a later stage of the project, other users came into the requirements: Staff. This user represents the functionary on the room floor whose role will be critical to report the occurrence of classes to the HR Section. If some professor misses a class, the functionary must use the system to

register it. These users are also allowed to get basic information from the system; mainly, related to room occupation.

At last, as the system needs users, and there are different types, another (super)user was identified. The portal administrator (Admin). This user will handle the maintenance of users' records and credentials.

The role of each actor was identified and its set of functionality defined according to requirements elicitation.

The Use Case List contains the following functionality: (a) Manage (create, delete) Room Reservation – Professor and Responsible (with different levels of permission); (b) List Room Reservation - all users; (c) Register faults; (d) Manage Rooms available and its resources (Responsible); (e) Import Timetables and set the semester dates (Timetable committee) and; (f) Manage Users.

The Figure 1 shows the Use Case Diagram produced based on this list of functional requirements.

Tools were selected considering the availability and the skills of the main contributors for this work to be concluded. Star UML[1] (version 5.0.2.1570) was the tool used to build all the UML diagrams.

Using a word processor, with a predefined template, descriptions of each use case were made to have necessary documentation for the next step – construction.

Along with this task of describe the Use Cases, User Interface design was made to get both components consistent: user interface and the action behind it. The Activity Diagram (Figure 2) was created to help understanding the circuit of actions.

[1] Star UML is an Open Source product made available at http://sourceforge.net/projects/staruml/

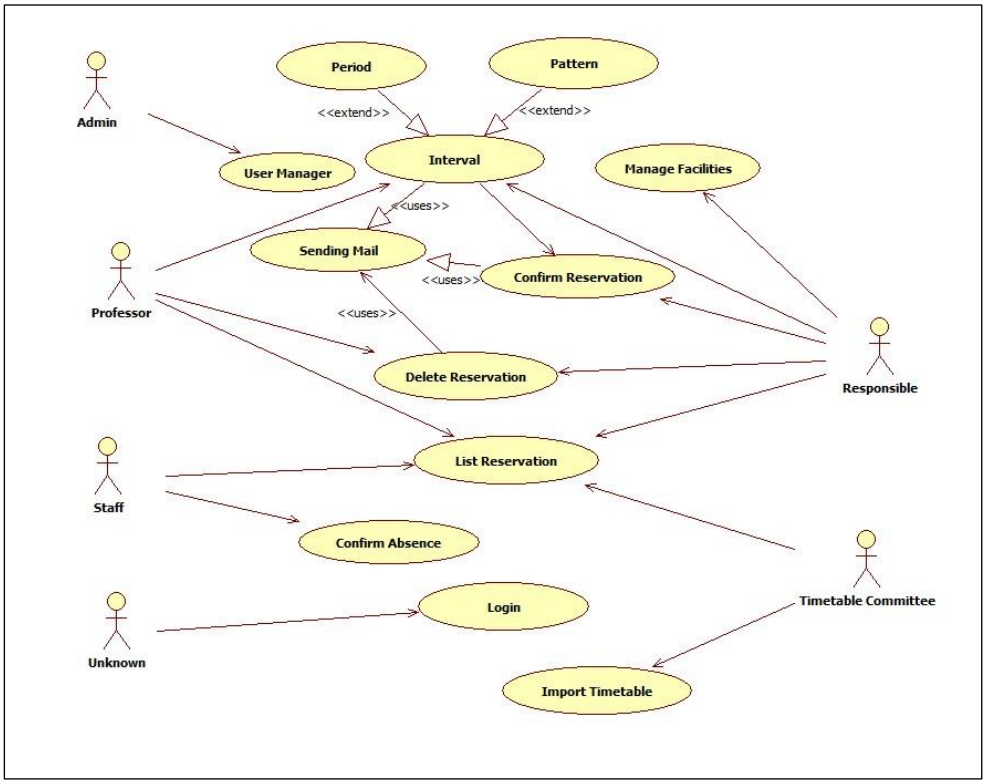

Figure 1 – Use Cases Diagram

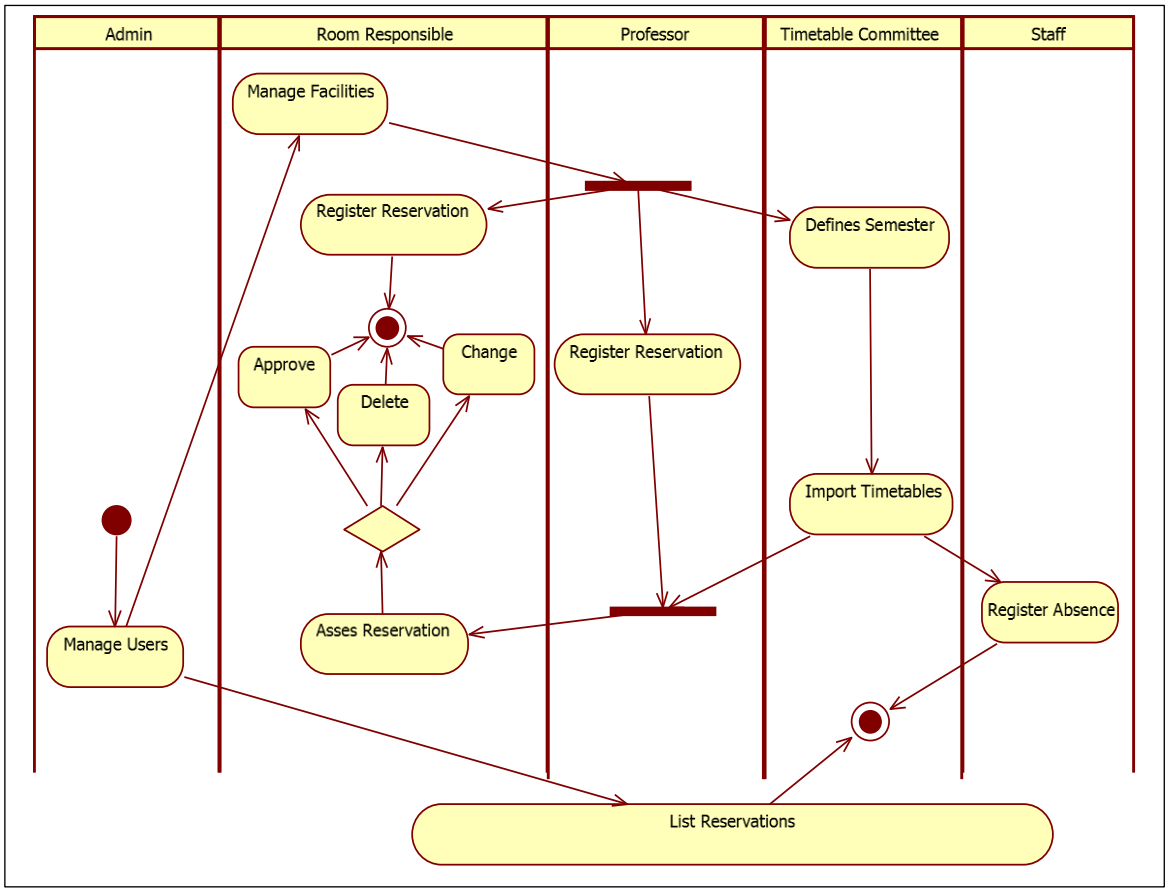

Figure 2 – Activity Diagram

Before the construction of the prototype, the UI design was made by hand (as shown in the Figure 3) and discusses with the users.

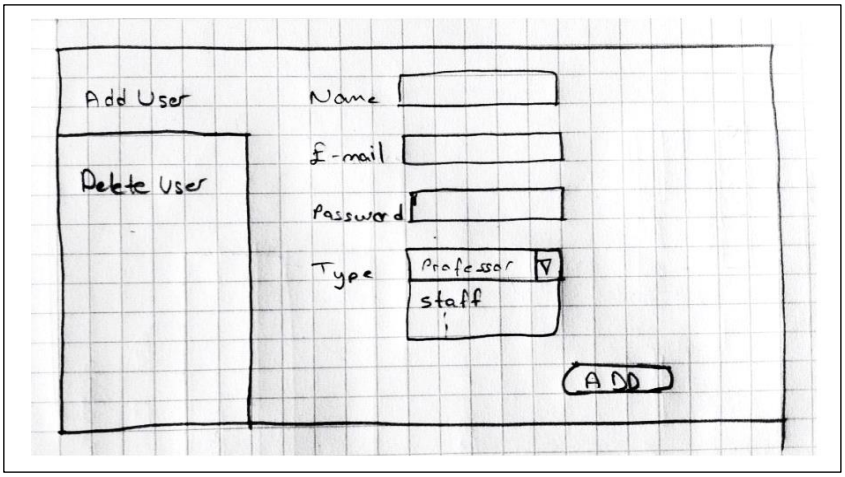

Figure 3 – Sketch of User Interface (User Management)

Then, the prototype was built, and the construction of this web portal enables the preview, more in detail, what the system will be at the end of the project. Figure 4 shows one example of it.

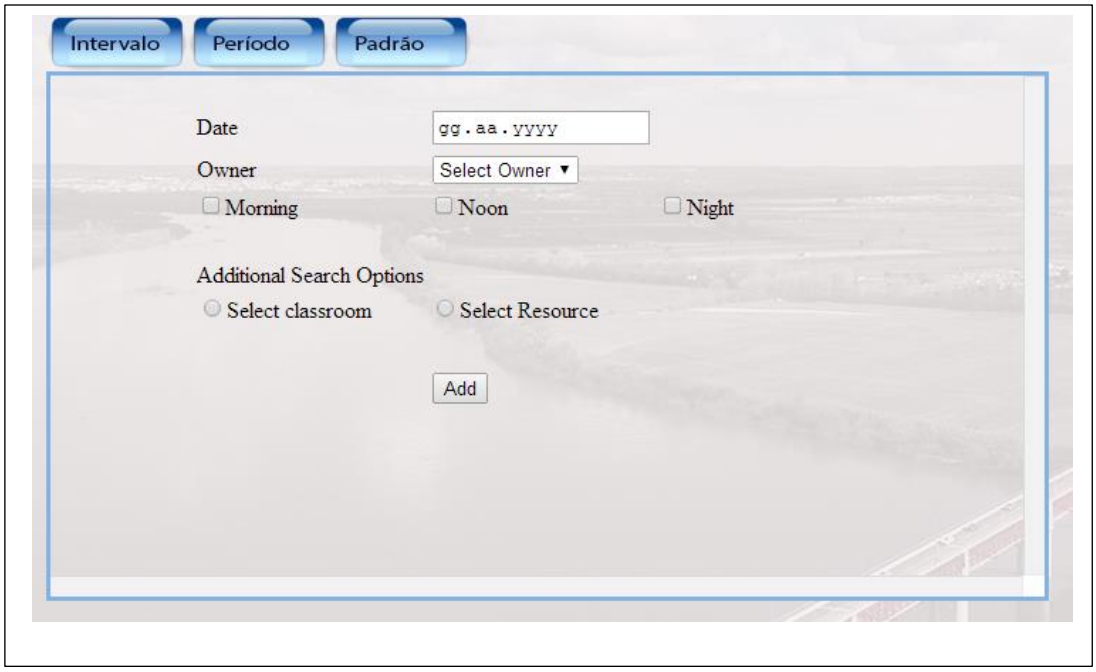

Figure 4 – Sample of Prototype's User Interface (Reservations Management)

At the Conception phase of the methodology, a Class Diagram was drawn (as shown in Figure 5) and the correspondent Database Schema was derived from it. The process that lead to database schema from the Class Diagram was performed according to the method described by Naiburg and Maksimchuk (2001).

During the development phase, no framework was used. All the code was developed using PHP (Vikram, 2009), CSS[2] (Holzschlag, 2003; Olsson, 2008) and javascript (Quigley, 2010), following the rules of object oriented development.

The operation of the system will be dependent on a Web server that supports PHP. The option was using the XAMPP[3] (Dvorski, 2007). As the system uses email to make the communication between users, instead of developing specific code, a public class was used – PMPMailer[4].

The integration is based on the asynchronous exchange of data through the importing of a file containing the data about timetables, structured using comma separated values (CSV[5]). This is the fundamental data for the room occupation at school, and will be provided through a CSV file, obtained from *Timetables Committee*.

All user interfaces are written in English. However, the system is prepared to change the idiom, based on a language file. During the deployment of the system, this files have to be changed according to the language adopted by the users.

The system may be tested at http://noodle.sytes.net:8088/cramp. The administrator credential are the following - username: “sisgersal@esg.ipsantarem.pt” and password: “admin”.

5 CONCLUDING REMARKS AND DISCUSSION

This paper presented the CRAMP – Classroom and Resources Availability Management Portal, through a description of its conceptual and technological bases, highlighting its features and the process that was used to construct it.

Regarding the conceptual solution, this was achieved by the application of agile software development, resulting in a functional system that is starting now to deliver the aimed results.

[2] http://www.w3.org/TR/CSS2/
[3] https://www.apachefriends.org/pt_br/index.html
[4] https://code.google.com/a/apache-extras.org/p/phpmailer/
[5] http://tools.ietf.org/html/rfc4180

Besides the functionality objectives, one other goal was achieved: the small dependency on technologies needed to make the exploration of the system. The almost zero costs of maintaining the system operational, because there are additional systems that use the same resources, makes this a system with high level of return.

In fact, it only takes an operating system like *Microsoft Windows*, and the lightweight, portable, easy to install and maintain, XAMPP server system. There is no need for an additional database engine because XAMPP has all that is needed.

From the perspective of the user, a *Web Browser* is the only piece of software required to use the system.

Finally, the future work related to this system is already under study and will lead to a construction of a mobile application. This app should allow users to access the system without any supporting layer (which is, without the need of a browser): just using a smartphone or tablet PC with the mobile application installed.

6 REFERENCES

Boronczyk, T. and Martin E. P. (2008). *PHP and MySQL – Create-Modify-Reuse*. Wiley Publishing.

Dvorski, D. (2007). *Installing, Configuring and Developing with XAMPP*. Available on-line http://dalibor.dvorski.net/downloads/docs/InstallingConfiguringDevelopingWithXAMPP.pdf

Hevner, A., March, S., Park, J. and Ram, S. (2004). *Design Science in Information System Research*. MIS Quarterly Journal, Volume 28, Issue 1, March 2004, Pages 75-105. http://misq.org/design-science-in-information-systems-research.html

Holzschlag, M. (2003). *Cascading Style Sheets. The Designer's Edge*. Sybex, Inc.

Jacobson, I., Booch G. and Rumbaugh, J. (1998). *The Unified Software Development Process*. Addison Wesley Longman.

Kroll, P. e Kruchten P. (2003). *The Rational Unified Process Made Easy*. Addison Wesley.

Naiburg, J. and Maksimchuk, R. (2001). *UML for Database Design*. Addison Wesley.

O'Docherty, M. (2005). *Object-Oriented Analysis and Design: Understanding System Development with UML 2.0*. England: John Wiley & Sons Ltd.

OMG (2003). *UML 2.0 Superstructure Specification*, ptc/03-08-02, available at http://www.omg.org.

Olsson, T. and O'Brien, P. (2008). *The Ultimate CSS Reference*. SitePoint Pty Ltd, Australia.

Orlikowski, W.J. and Iacono, C.S. (2001). *Research Commentary: Desperately Seeking the 'IT' in IT Research – A Call to Theorizing the IT Artifact*, Information Systems Research (12:2), June 2001, pp. 121-134.

Quigley, E. (2010). *JavaScript by Example*. (2nd ed.). Prentice Hall.

Schmuller, J. (2004). *Teach yourself UML in 24 hours*, (3th ed.). SAMS publishing.

Vikram, V. (2009). *PHP – A Beginner's Guide*. McGraw-Hill, Companies.