DESIGN AND DEVELOPMENT OF AN UNDERGRADUATE COURSE ON INTERNET APPLICATIONS BASED ON AN INTEGRAL PEDAGOGICAL APPROACH

Camilo Jiménez and Jorge Villalobos
Systems Engineering and Computing Department, University of Los Andes, Bogotá, Colombia

Keywords: Active Learning, Education, Pedagogical Model, Rich Internet Applications, Web Application Development.

Abstract: Several aspects such as the large amounts of knowledge that can be found in the Internet applications field, the rapid and continuous evolution of Internet, and methodological problems that have been commonly reported in computer programming courses, challenge the designing and development of Internet application courses. In this paper, we present an integral pedagogical approach to design and teach such courses, considering all of those aspects by means of: (1) The definition of a balance among several thematic axes, and the generation of high-level programming skills; (2) a structured set of learning resources along with tools that facilitate the management and evolution of them; and (3) a pedagogical active learning model that helps instructors teach in a systematic way.

1 INTRODUCTION

The importance of Internet Applications has been growing rapidly during the last ten years. Nowadays, these applications are recognized to be the best way for doing business efficiently and quickly and, for this reason, they are considered as a key IT qualification in any computer scientist (Dörge, 2008). Considering this situation, it is essential for computing programs to design and teach courses on Internet applications within their curricula. Nevertheless, aspects inherent to the rapid evolution of the Internet field and the current situation of some computing education environments make this task challenging.

The first of these aspects is the large amount of knowledge that can be found around the field. The concepts and technologies in this area are so vast that designing and teaching a course requires great knowledge from instructors and the ability to select coherent topics that can be presented (Lee, 2003). This is also related to another problem: Confronted to such amount of knowledge, courses on Internet Applications have been usually designed following roadmaps around a programming language or a limited set of technology elements. The consequence of this is that the courses, and many of the books around learning Internet Applications, are based on a review of the syntactical structures of the programming language or technology elements. This way, other aspects such as usability, sociability or even architectural aspects, which are essential to the construction of current Internet applications, are viewed superficially or sometimes ignored.

The second aspect is the rapid evolution of Web technologies. Each current available technology is constantly changing, requiring a significant amount of effort from instructors to keep up-to-date with the latest technology. Similarly, course materials are also forced to be updated continuously. This augments the effort from instructors even more since they have to design, develop, and maintain them.

Finally, another aspect is the set of methodological difficulties that computer programming courses have reported for the last twenty years (Woodley, 2007). Particularly, students have been the focus of recurrent issues relating to their frustration, and instructors have shown a lack of implanted feedback mechanisms. Consequently, Computing departments have shown difficulties when managing and supporting knowledge generated in these courses, which generally depend heavily on the instructor abilities rather than a clear pedagogical model (Villalobos, 2009a).

Jiménez C. and Villalobos J. (2010).
DESIGN AND DEVELOPMENT OF AN UNDERGRADUATE COURSE ON INTERNET APPLICATIONS BASED ON AN INTEGRAL PEDAGOGICAL APPROACH.
In *Proceedings of the 2nd International Conference on Computer Supported Education*, pages 426-432
DOI: 10.5220/0002794904260432

Several tools and strategies (Gearailt, 2002; Kölling, Quig, Patterson & Rosenberg, 2003; O'Kelly, Gibson, 2006) along with different approaches to design and teach courses on Internet Applications (Yue, 2004; Lee, 2003; Hu, 2004) have been developed but none of them seem to tackle the complete set of challenges associated to the three aspects mentioned above. In this paper, we present an *integral pedagogical approach* that deals with all these challenges by means of: (1) The definition of a balance among several thematic axes, and the generation of high-level programming skills; (2) a structured set of learning resources along with tools that facilitate the management and evolution of them; and (3) a pedagogical active learning model that helps instructors teach in a systematic way.

This approach is part of a larger project named Cupi2 whose objective is to search for new ways to learn/teach computer programming. Particularly, we adopted, customized, and extended some elements from a previous experience in which we defined an approach to design and develop the basic computer programming courses CS0, CS1, and CS2. This approach was successfully evaluated (Villalobos, 2006; Villalobos, 2009a,b) and it has conducted Cupi2 to be recognized by important regional institutions in two different occasions. In the first occasion, Cupi2 was awarded the 2007 Colombian Informatics Award by the Association of Colombian Computer Engineers (ACIS), based on the quality of its learning objects and its academic impact in more than 30 universities in Colombia. In the second occasion, Cupi2 obtained the first place in the 10th prize of Educational Informatics 2009 by the Iberoamerican Network of Educational Informatics (RIBIE), based on its academic and research quality, its social incidence, and the number of students and faculty members benefited from it.

Following the integral pedagogical approach we propose in this article, we designed and developed an Internet applications course called Rich Internet Applications (RIAS). This course has been taught in a yearly basis as an elective course in our Computing program for three years, showing promising results.

The structure of this paper is as follows: In the next three sections we discuss around the three aspects that challenge the design and development of an Internet applications course, along with the solutions that we propose within an integral pedagogical approach. Then, we analyze some metrics in order to validate our approach. In the next section, we discuss some related works, and finally the conclusions of the paper are presented.

2 EQUILIBRIUM OF THEMATIC AXES AND HIGH-LEVEL PROGRAMMING SKILLS

Due to the existence of large bodies of knowledge around Internet Applications, selecting the concepts to teach RIAS was a very challenging task. The first thing we realized was that trying to cover as much knowledge as possible based on a pass-over of the syntactical structures of the different Web programming languages and technologies was inadequate. In this classic approach, students tend to ignore other important aspects to Internet applications giving much more importance to Web programming languages and technology elements than the way to integrate them into the process of building an application. At the end of such courses, students have the impression that they learnt to use a lot of languages and technology elements but they are not explicitly conscious about the abilities they should have generated in other axes. This usually generates frustration in students since some cases they believe that the particular programming language or the set of technology concepts is not going to be useful enough in their professional life.

Contrary to this classic approach, we aim at finding a *balance among several thematic axes*. This means that instead of focusing in specific syntactical structures or technology concepts, we want students to understand that building an Internet application requires mastering several concepts from domains like architecture modelling, usability and sociability, security, programming tools, technology, etc. With this, we do not intend to cover all the large bodies of knowledge around Internet applications. On the contrary, our intention is to define enough conceptual fundamentals that can be complemented by the generation of *high-level skills*; this means, abilities to use this knowledge effectively to solve problems in similar contexts (different technologies for example). Such skills include understanding and abstracting a problem, decomposing it, modelling a solution, analyze it, etc. The complete set of high-level programming skills is illustrated in (Villalobos, 2009b).

2.1 Thematic Axes

The first thematic axis in RIAS is the *architectural axis*. It refers to the ability to abstract relevant information from a problem and identify the necessary structures in architectural models that can represent a solution to that problem. This axis also

refers to the ability of understanding how an Internet application works from different viewpoints such as functional, security, concurrency, data, and deployment. In the course, we introduce, and in some cases reinforce, concepts such as client, server, protocol, session, concurrency, thread, component, application, control, data flow, communication methods, etc.

The second thematic axis is the *Usability and Sociability axis*. It refers to the ability to understand and use concepts around usability and sociability that are defined within the Web 2.0 trend (Anderson, 2007). These concepts become essential in RIAS due to their great recognition and applicability in nowadays Internet applications. This axis also refers to the ability to identify patterns as well as to analyze inherent standards and best practices related to these two aspects. In the course, we introduce concepts such as usability, virtual communities, community governance, patterns, collaboration, collective intelligence, Web 2.0 applications, user experience, etc.

The third axis is the *Security, Ethics, and Privacy axis*. It refers to the ability to identify vulnerabilities in current Internet applications at various levels of the application from an architectural perspective. This axis also refers to the ability to propose strategies and guides that lead to best practices around security, and to understand ethics and information management behind Internet applications. In the course, we introduce concepts such as secure protocols, attacks, password encryption, firewall, privacy statements, cross side scripting, SQL injection, faults, fails, system recovery, etc.

The fourth axis is the *Frameworks and Technologies axis*. It refers to the necessary technological elements that support the construction and deployment of an Internet application. In this course, we use development tools such as Eclipse and Adobe solutions, application containers such as JBoss and Glassfish, and frameworks such as Pushlets, JSF, Google APIs and libraries, etc.

The fifth axis is the *Languages and Programming axis*. It refers to both the different ways to express the architectural structures of an Internet application into a program, and the set of strategies and guides that help in this process. In this course, we introduce and reinforce languages such as Java, Javascript, HTML, XML, etc.

The sixth axis is the *Testing axis*. It refers to the ability to understand testing from two perspectives: Firstly, as a way to automatically find functional faults and fails of an Internet application; and secondly, as a way to evaluate and tune quality attributes such as performance and usability. This axis also refers to the ability to define, model and construct testing scenarios and test cases considering the two different perspectives. In this course, we introduce concepts such as unit testing, load balancing, functional testing, usability evaluation, application tuning, etc.

3 COURSE MATERIALS

Defining the appropriate materials for RIAS considering the vast and rapidly changing environment around Internet applications represented a very challenging issue for several reasons. First, reference books around Internet applications are very diverse in scope and complexity levels. For this reason, it is very unlikely to find one book that fits within the complete set of concepts defined in the thematic axes. On the other side, finding several books, each one specialized in certain group of concepts, would confuse students since methodologies and learning activities defined in each one of them are different and various. Second, public tutorials and learning objects available on the Web usually lack of an academic validation and most times they are not adequately aligned with the development of programming skills. Additionally, some of them are subject to copyright norms and others are simply not continuously updated.

Considering the problems finding suitable existing course materials for the course, we decided to create our own set. Particularly, we defined *learning objects* focused on the generation of high-level skills, a *virtual book* for covering the complete set of concepts defined in the thematic axes, and two corresponding collaborative tools that facilitate the management and evolution of both the learning objects and the contents of the virtual book.

3.1 Learning Objects

According to the pedagogical active learning model explained in the next section, different activities in the course should be supported by a considerable amount of learning objects that engage students as the main role in the learning process (Villalobos, 2009a). We categorized these important elements of the course material as: Examples, laboratory workshops, working sheets, tutorials, animations, demos, discussion workshops, mind maps, exams, and interactive learning objects (Villalobos, 2009b).

To facilitate the planning, management and evolution of learning objects we created a profile for RIAS within the Cupi2 online community (Villalobos, 2009a). This community promotes the generation of collective knowledge around learning objects and also offers a continuous support to instructors involved in active learning approaches. In order to do this, instructors are allowed to create learning sequences and blogs online, facilitating the planning of class sessions and their association with particular learning objects, and allowing the management of the records of their own experiences.

3.2 Virtual Book

The virtual book is a structured set of contents derived from the compendium of lecture notes and slide presentations that have been generated by instructors in the last three years. It not only covers the fundamental concepts defined in the thematic axes of the course, but it is also focused on the generation of high-level skills, and grounded in the pedagogical active learning model explained in the next section.

Currently, the contents of the book are managed in a wiki with limited access to students and instructors. Nevertheless, we are working on the nurturing of an online community around the contents of the book that promotes the generation of collective knowledge around it. This way, the contents evolve in a collaborative environment facilitating the inclusion of concepts from different thematic axes. The virtual book is out of the scope of this paper and the beta version of the online community around its contents is planned to be released next semester.

4 PEDAGOGICAL ACTIVE LEARNING MODEL

Since 2004, the Cupi2 project (http://cupi2.uniandes.edu.co) has been working on the definition of different active learning pedagogical approaches to support computer programming teaching/learning. Specifically, these approaches have managed to overcome the well-known methodological problems around basic computer programming courses, such as CS0, CS1, and CS2, (Villalobos, 2009a) by means of the definition of a constructivist pedagogical foundation. Each foundation is based on four main components that engage students as the main role in the learning process, and establish a well balanced solution to these problems (Villalobos, 2009b).

4.1 Active Learning

The first component is Active Learning. According to this, teachers must act less as instructors but more as promoters of activities that ensure the generation of relevant skills for problem analysis and solution modelling (Villalobos, 2009a). This changes the *class and laboratory structures* of courses completely and the challenge of instructors becomes to define an adequate set of *scenarios* where students are motivated to work, with clear objectives related to evaluation.

Class and laboratory structures in RIAS are categorized in lecture and practice sessions. For every two lecture sessions, there is a practice session in the laboratory. This lab session aims at putting the concepts reviewed from the lectures in practice using technology concepts and software tools.

Each of the class sessions in active learning environments must be supported by a set of scenarios that engages students in activities beyond listening to a lecture. These scenarios must be supported by appropriate learning objects with different objectives (Villalobos, 2009b). In RIAS, we have defined and developed more than 100 learning objects. All of them are managed in the Cupi2 Community within the corresponding course profile.

4.2 Problem-based Learning

The second component is Problem-based Learning (PBL). In PBL, students should learn through facilitated problem solving where concepts are bound to specific issues and explained through their relation with a specific part of the *problem* (Villalobos, 2009a). This offers the potential to help students develop flexible understanding and high-level skills (HMelo-Silver, 2004), and it increases their motivation since they are confronted with problems that reflect real-world challenges.

In RIAS, each level begins with the presentation of a given problem to be solved in groups of three students. Particularly, these problems correspond to specific issues that can be solved by completing a specific part of a big Internet application. This way, we intend to keep the balance among the different thematic axes since students are constantly faced with the “big picture” of these applications.

4.3 Incremental Learning

This component is essential in the pedagogical foundation considering the large number of concepts in computer programming courses. These concepts must be introduced in a gradual way so that the balance between their complexities can be preserved. To accomplish this, we define the notion of *levels*. Each of these levels is defined to (1) introduce or reinforce some knowledge using learning objects in different class sessions, and (2) generate or strengthen of a set of skills defining a particular problem to solve. The relationship between levels and other elements in the foundation is depicted in Figure 1.

In RIAS, we defined three levels that incrementally introduce concepts around Internet applications. As an example, the first of them focuses on the user and his behaviour. In this level we introduce concepts from different axes related to usability, sociability, content and presentation, and fundamental architectural styles to build Internet applications. The problem associated with this level is intended mainly to generate skills in the understanding, abstracting, and analysis of the architectural aspects around current Internet applications. This way, students are challenged to analyze and design an online community, and implement explicit functional requirements in a simple Internet application that supports such community. This problem continues to grow in complexity each level. The idea is to reuse the implemented solution and incrementally augment it without losing the "big picture" of the application. We have realized that letting students propose and define a community of their preference motivates them in the solution of the problem. In particular, four groups of students have continued working in their communities as venture projects after the end of the course.

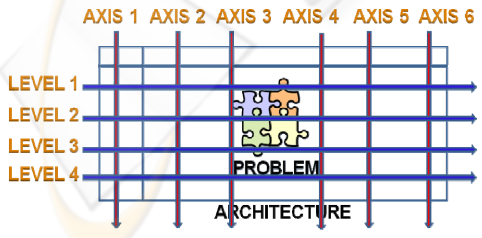

Figure 1: Levels, thematic axes, and problems.

4.4 Learning by Examples

The last component is Learning by Examples. This component enforces the development of examples to enrich the learning process. These examples are transversal methodological elements in our approach and they must show the applicability of best practices and common solutions (Villalobos, 2009a).

In RIAS, we have developed two examples per level. These examples are internet Applications that support two online communities in an incremental approach. An important aspect of the development of these examples is that a *standard conceptual model* for them was needed. Otherwise, students would become confused when changing between levels since it would be difficult to localize concepts again in another example.

The standard conceptual model in RIAS considers examples as applications constructed following a client/server basic style: A client that sends requests to a server through a communication channel using a standard protocol, and a server that attends several of these client requests. In addition to this, examples include an attractive graphical interface that considers a set of usability patterns, a common architectural document, a set of test cases, a well-documented code, and a strategy to solve the problem.

5 METRICS AROUND THE COURSE

RIAS is the first course focused on Internet applications that has been included in the curriculum at our University. In the first time we taught RIAS, 35 students took the course. The second, this quantity grew by a 100%, making necessary to teach two sections of the course. After this, the number of students has normalized, and we now usually teach two sections that totalize between 50 and 65 students every time the course is taught. Among other 10 established and older elective courses, RIAS is the one with the most number of students taking the course. This represents a great success considering we began teaching the course just three years ago.

The course has been always subject to quantitative and qualitative evaluations by the Computing department. In quantitative evaluations, the department measures the satisfaction of students in terms of the overall course, the course materials, and the pedagogical methodology. In these three years, the course has been qualified with a satisfaction average of 82%, 84%, and 82% in terms of the corresponding criteria. In qualitative evaluations, the department asks about the very positive aspects of the course as well as the negative ones. In summary, very positive comments are found

about the methodology and the appropriateness of learning objects in several class sessions. Students appreciate the great quantity of course material available for them during class sessions and when they are studying at home. On the other side, negative comments about the contents of the course have been reported. In particular, some students are not satisfied following the course with a Java perspective and others want to augment or detail the scope of the topics covered. This is an issue that we are trying to solve currently, and our hypothesis is that we have to reinforce and make explicit the generation of high-level skills, and develop learning objects in other technologies following the examples' conceptual model defined within our approach. In addition, we are going to let students decide their own technology when solving problems. This way, we can check the learning process of different groups when solving problems in different technologies and measure exactly the impact that this particular issue has in the learning process.

6 RELATED WORK

Several related works that aim at finding approaches to design and develop courses on Internet applications have been proposed. Some of them have been oriented to technology. This is the case of (Yue, 2004) where Kwok-Bun Yue et al propose the design and evolution of an Internet applications course based on their eight years of experience in teaching such course. Particularly, they propose a syllabus that covers an ordered set of available Web technologies and establishes a mechanism to select them based on their popularity and maturity. Together with this mechanism, they propose an incremental approach to include new emerging Web technologies in the course as instructors gain expertise with them and the technologies augment its maturity and support. The main problem with this approach is the high risk of generating frustration in students since the syllabus structure is not explicitly conscious about the abilities they can generate in other axes. This suggests that building Internet applications is all about learning a programming language, leveraging the importance of the field and in some cases decreasing students' motivation. Another problem is the high dependence on the Web technologies. Considering the number of current different technologies and the rate at which they grow currently, there will be always a large set of relevant technologies that will be left aside because they do not fit into one semester. Again, students will feel that they are not covering the complete set of subjects, generating frustration in them. In our case, our approach is totally different. We focus in the balance of several basic fundamental concepts and the generation of different skills to use those concepts effectively in problem solving. This problem solving could include the adoption of new Web technologies or the usage of new languages.

Other related works have been oriented to concepts. This is the case of (Lee, 2003) and (Klassner, 2000) where an approach based on the teaching of enough fundamentals to solve the challenges around teaching/learning Internet application courses is described. The premise of these approaches is that students should gain the necessary abilities and core conceptual concepts in the course so that they can effectively apply them in a productive environment. In particular, we follow the same premise. The difference with them is that they lack of an integral pedagogical model supported by collaborative tools that facilitates the management and evolution of course material as well as the teaching of the course by means of an adequate pedagogic methodology.

Another set of related works have been oriented to the use of tools to facilitate teaching. This is the case of (Hu, 2004) where a teaching tool called ETE is described. Several dimensions were not covered in that article, suggesting a lack of pedagogic elements within the course that are needed in order to solve the challenges around the teaching/learning Internet application courses. In particular, we believe that the solution cannot be found in just one element. On the contrary, these challenges need to be solved from different dimensions within an integral methodological approach that considers different dimensions such as students, instructors, Computing departments, pedagogic methodology, tools, course materials, etc.

7 CONCLUSIONS

In this article, we have presented an integral pedagogical approach to design and teach a course on Internet applications. This approach is part of a larger project called Cupi2, and it adopts, customizes, and extends some elements from a previous pedagogical experience in CS0, CS1, and CS2 courses.

Our approach considers three well-known aspects that challenge the task of designing and teaching such a course. First, large bodies of knowledge around Internet applications are

supported by a balance among several thematic axes and the generation of high-level programming skills. This way, students can acquire core fundamental knowledge from six different thematic axes and generate skills to apply that knowledge effectively in similar situations. Second, rapidly changing technologies are supported by a set of more than 100 learning resources along with tools that facilitate the management and evolution of them. Particularly, different types of course materials were developed and are currently managed in an online community that facilitates their evolution. And third, classic methodological difficulties in computer programming courses are supported by a pedagogical active learning model that helps instructors teach in a systematic way based on levels, problems, and examples.

Students have evaluated the course and although they report good and promising results, we have to improve some issues that were identified. We will also continue with the developing of learning objects, which represent one of the most important course's strengths according to students, as well as with the nurturing of the online community around the virtual book of the course.

REFERENCES

Anderson, P., 2007. What is Web 2.0? Ideas, Technology and Implications for Education. In *JISC Technologies and Standard Watch*.

Dörg, C., Schulte, C., 2008. What are Information Technology's Key Qualifications?. In *ITiCSE'08, 13th Annual Conference on Innovation and Technology in Computer Science Education*. ACM.

Gearailt, A. 2002. Teaching with Java: Using Java to increase active learning in programming courses. In *Proceedings of the inaugural conference on the Principles and Practice of programming*, Dublin, Ireland, June 2002.

Hmelo-Silver, C., 2004. Problem based Learning: What and How do Students Learn?. In *Educational Psychology Review 16 (3)*.

Hu, J., Hu, G., 2004. Teaching Web Design and Web Programming Using an Enhanced Teaching Environment. In *Proceedings of the Mid-South College Computing Conference*. Lousiana. USA.

Kölling, M., Quig, B., Patterson, A. & Rosenberg, J. (2003). The BlueJ system and its pedagogy, In *Journal of Computer Science Education*, Special issue on Learning and Teaching Object Technology, Vol 13, No 4.

Klassner, F., 2000. Can Web Development Courses Avoid Obsolescence?. In *ITiCSE'00, 4th Annual Conference on Innovation and Technology in Computer Science Education*. ACM.

Lee, Arthur., 2003. A Manageable Web Software Architecture. In *ITiCSE'03, 7th Annual Conference on Innovation and Technology in Computer Science Education*. ACM.

O'Kelly, J., Gibson, J. (2006) RoboCode & problem-based learning: a nonprescriptive approach to teaching programming. In *Annual Joint Conference Integrating Technology into Computer Science Education. Proceedings of the 11th annual SIGCSE conference on Innovation and technology in computer science education.*

Villalobos, J., Casallas, R., 2006. Teaching/Learning a First Object-Oriented Programming Course outside the CS Curriculum. In *10th Workshop on Pedagogies and Tools for the Teaching and Learning of Object Oriented Concepts - ECOOP (European Conference on Object-Oriented Programming)*.

Villalobos, J., Calderón, N., Jiménez, C., 2009a. Cupi2 Community: Promoting a Networking Culture that Supports the Teaching of Computer Programming. In *CSEDU'09, 1th International Conference on Computer Supported Education*. Portugal.

Villalobos, J., Calderón, N., Jiménez, C., 2009b. Developing Programming Skills by Using Interactive Learning Objects. In *ITiCSE'09, 14th Annual Conference on Innovation and Technology in Computer Science Education*. ACM.

Woodley, M., Kamin, S., 2007. Programming Studio: A course for improving programming skills in undergraduates. In *Proceedings of the 38th technical symposium on computer science education*. USA.

Yue, K., Ding, W., 2004. Design and Evolution of an Undergraduate Course on Web Application Development. In *ITiCSE'04, 09th Annual Conference on Innovation and Technology in Computer Science Education*. ACM.