2016 IEEE 29th International Conference on Software Engineering Education and Training

Generating Software Engineers by Developing Web Systems: A Project-Based Learning Case Study

Patrick Letouze
Department of Computer Science
Universidade Federal do Tocantins
Palmas, Brazil
Email: patrick.letouze@gmail.com

J. I. M. de Souza Júnior
Department of Computer Science
Universidade Federal do Tocantins
Palmas, Brazil
Email: jhoseju@gmail.com

Valéria Martins da Silva
Program of Systems' Engineering and Computation
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brazil
Email: valeriamartinsdasilva@gmail.com

***Abstract*—The novelty proposed in this work regarding teaching and training of software engineering is about how the Project-Based Learning approach is performed for developing a web system for managing academic projects. The supporting project is a real-life problem. Its development was planned in four phases: prototype, code refactoring, release-to-manufacturing and deployment. We employed some strategies such as role-playing, a software house simulation, and the Model-View-Controller pattern combined with Evolutionary Acquisition and the Interdisciplinary Research Project Management framework.**

***Keywords*-Graduate program management, project-based learning, project management, student-centric, web system.**

I. Introduction

The implementation of Project-Based Learning (PBL) to the teaching and training of software engineering has been successfully applied for undergraduate students. For example, it has been used for teaching software architecture [1] and in a first programming course [2]. Actually, its use is included in the recommended sets of best practices for computing curricula such as the ACM/IEEE's CS2013 [3]. A clear testimony of its application in undergraduate level is given by Gary [4]:

> For the past decade at Arizona State University (ASU), we've implemented PBL as the central feature of the Bachelor of Science in Software Engineering... we have the responsibility not only to equip our students with the technical skills they need to start a career but the ability to apply, evolve, and practice those skills throughout their lifetime. While many forms of learning have their place, sustained PBL provides students with these more durable benefits.

In this work, we propose a PBL approach that is simultaneously curricular and extra-curricular. Using this approach, we developed a web system for managing academic projects. Our PBL project represents a real-life problem, the client is the University where it was developed, that is, the *Universidade Federal do Tocantins* (UFT - Federal University of Tocantins), Brazil. Our PBL setting includes role-playing and a software house simulation. The software development process was divided in four Phases: prototype, code refactoring, release-to-manufacturing and deployment. The project initiated in the very beginning of 2010 and its deployment was concluded at the end of 2015.

First, in phase 1 - prototyping, after a lot of debate, the students chose the Unified Process as the software implementation methodology and the MVC architecture pattern. This choice was strongly influenced by the bibliography adopted in the course [5]. All tools and techniques were chosen by students, and in particular, JAVA was a choice derived from the previous disciplines.

In the second phase, the complete code-refactoring [6], the students were told to continue the system's development. Now the only professor continuing the project, and the first author of this paper, did not reveal its intention of a complete code-refactoring of the system. The students faced the difficulties of continuing the develpoment of a system poorly documented and poorly coded. After much effort they asked for a meeting where they proposed the complete code-refactoring. Now the prototype served only as a support for requirements analysis. Again, the students had the opportunity to choose all tools and techniques, but regarding the software development methodology, the students decided to use a new approach. Hence, after some evaluation, they decided to use the Evolutionary Acquisition Interdisciplinary Research Project Management (EA-IRPM) [7]. This choice was strongly influenced by the fact that this approach was published by the students' supervisor, professor Patrick Letouze, the first author of this paper. Out of curiosity, the students asked the professor to make a presentation of the approach, which induced them to choose it. Additionally, because of the decision of providing scalability for the system, so it could be a platform for all Graduate Programs of UFT, the system's architecture had to change.

In the third phase, release-to-manufacturing, the prorector of graduate studies and research requested new requirements to support other activities of the Universityafter, consequently new developments and some code-refactoring

2377-570X/16 $31.00 © 2016 IEEE
DOI 10.1109/CSEET.2016.11

were necessary. Here the MVC EA-IRPM approach was not only used as the development strategy, but though code-refactoring it has become the system's architecture itself [8], [9], then a new version of the system was published in [10].

The fourth and last phase, the system's deployment, started with the submission of source code for registration at *Instituto Nacional de Propriedade Intelectual* (National Institute of Intelectual Property), which was granted [11].

This PBL approach is not only hybrid regarding the curricular dimension. Because of 6 years of development, this approach was performed not in one semester or just one year, therefore not just with one class of students, nor in only one discipline. It was performed with different groups of students in two different PBL conditions. In every phase the students had an active role in choosing tools and strategies.

Hence, we share this case study experience in the remainder of this paper and its organization is as follows. In Section II, we present briefly the web system for academic project management problem. In Section III, the settings of our PBL approach are shown. The software development methodology applied in all phases except the first is presented in Section IV. The phases 1 to 4 are reported in Sections V to VIII: prototyping; code refactoring; release-to-manufacturing; and deploying. Finally, in Section VIII, we discuss the achievements and the lessons learned.

II. The History of the Web System for Academic Project Management

This work reports the development of a web system for academic project management. Its main objective is to support the educational demands of "Graduation Projects", such as planning and supervision, of *Universidade Federal do Tocantins* - UFT. It means that it was conceived to support both graduate and undergraduate programs.

The main non-functional requirements of the web system are the student-centric approach, the incorporation of project management concepts in the system's architecture, and to provide a simple framework for data acquisition and analysis of educational measures to help coordination purposes.

In the first phase the main concern was to manage academic projects in terms of documentation such as proposal, chapters and the final monography, all respecting project management concepts, student-centric approach and the structure of a University.

In the begining of the second phase, it had become clear that some mistakes had been done in the University structure, also the system was poorly documented and the source code was unnecessarily extensive and difficult to understand. Hence, a complete code-refactoring was the strategy adopted to first correct the system's architecture, second to optimize de source code, third to provide a better documentation for further developments, and fourth, to implement new features such as a student timeline and the concept of milestones from project management as metrics for monitoring the academic project evolution, graphs and the student performance in a Graduate Program Level.

In the third phase, the graphs had been code-refactored and the new focus was communication. The formal communication between student and supervisor was consolidated in the architecture using the concepts of evolutionary acquisition. Moreover, communication between student and supervisor, student and coordinator of the program, professor and coordinator of the program, were all accomplished in a social network structure. Now the system is simultaneously a social network and an academic project manager.

After the software registration, the deployment started and new requirements emerged demanded by the pro-rector of research and graduate studies, then a new module was developed to attend these needs.

III. Project-Based Learning

The learning strategy applied in this work was the Problem-Based Learning. According to Savery [12]:

> Problem-Based Learning is an instructional (and curricular) learner-centered approach that empowers learners to conduct research, integrate theory and practice, and apply knowledge and skills to develop a viable solution to a defined problem.

Actually, the strategy employed is also a Project-Based Learning pedagogy, because "the learning activities are organized around achieving a shared goal (project)" [12]. Alternatively, a definition for Project-Based Learning (PBL) is presented by Bender in [13]:

> Project-Based Learning is an instructional model based on having students confront real-world issues and problems that they find meaningful, determine how to address them, and then act in a collaborative fashion to create problem solutions.

The real-world problem to be solved was how to support educational planning and management of academic project-based programs. Specifically, as a meaningful problem to the students, the context of the problem was the discipline "Graduation Project" of the Computer Science curriculum. Besides, it should be applicable to a Master Science or Doctorate program. Hence, the solution was to develop a Web system by simulating a software house, where the students would role-play as development team members, that is, a collaborative software development project.

Bender in [13] presents a summary of language PBL that serves simultaneously as basis or essentials for a PBL approach. The words or concepts are:

- Anchor - the basis for posing the question that serves to ground the instruction in a real-world scenario.
- Artifacts - the items that represent possible solutions to the problem or aspects of the solution to the problem, role-play scenarios are included.

- Authentic achievement - represent the emphasis, the type of things professionals might expected to do in real-life.
- Brainstorming - this is a process students undergo to formulate a plan for project tasks.
- Driving question - the primary question that provides the overall goal of the project.
- Student voice and choice - it represents that students should have some say in project selection and statement of the essential question.

The novelty proposed here regards teaching and training of software engineering is that the Project-Based Learning approach is not performed in one semester or one year, nor it is performed with the same group of students, nor restricted to a discipline or class. The project that guides this work was planned in phases and its aim is a real web system to be used at least by the University where the development was executed. In Table I, we show each phase and its objective.

Role-playing in education intends to prepare students for a future performance and to improve abilities within a role. Hence, throughout the project, somehow a software house was simulated. Particularly, in the first phase, it was explicitly declared the simulation of a software house. The context of this phase was that a software house was simulated in the class of the discipline of "Systems' Development", where the students played the role of the development team. The PBL scenario of phase 1 is shown in Table II.

In the other phases, it was not emphasized the simulation of a software house, though it was informed that the project intended to provide them a similar experience to a development team in a software house. It was expected that the documentation from phase 1 would be incomplete and that the source code would be poorly comprehensible. For that reason, it was planned a second phase where a complete code refactoring would be performed. Thus, we invited a few students for the second phase without revealing our expectations, and only providing the source code for them to study the project, but now revealing that the true intention was to develop a web system that would be employed at the University level for supporting all Graduate programs. The PBL scenario of these phases is presented in Table III.

Table I
SYSTEM'S DEVELOPMENT PHASES

Phase	Objective
1: Prototype	The system's development intends to validate its main functionalities.
2: Code Refactoring	The system's development intends to have the functionalities operational.
3: Release-to- Manufacturing	The system's development intends to have a stable release ready to be delivered or provided to the customer.
4: Deployment	The intention here is to execute all the activities that make the system available for use.

Table II
PBL SCENARIO FOR PHASE 1 OF THE PROJECT

PBL Scenario for Phase 1: Protoype	
Concept	**Description**
Anchor	The problem presented to the students was the need of a web system for managing academic projects, especially in the context of the discipline "Graduate Project" of the curriculum of Computer Science.
Artifacts	The modules required for the web system, especially in terms of Model-View-Controller.
Authentic Achievement	The students would use the system when they were enrolled in the discipline "Graduate Project".
Brainstorming	Brainstorming This process was performed almost every week during classes. When it was required a major project decision, the brainstorming happened with the participation of all students. Minor decisions regarding one module only were made through brainstorming with the responsible team.
Driving Question	Is a web system for managing academic projects based on project management concepts a valuable asset for improving the success and quality of "Graduate Projects"?
Student Voice and choice	The students have helped to choose the tools and techniques for the system development.

Table III
PBL SCENARIO FOR PHASES 2, 3 AND 4 OF THE PROJECT

PBL Scenario for Phases 2, 3 and 4:Code Refactoring, Release-to-Manufacturing and Deployment	
Concept	**Description**
Anchor	The problem presented to the students was the need of a web system for supporting the management of graduate programs.
Artifacts	The modules required for the web system, especially in terms of Model-View-Controller.
Authentic Achievement	The development of the system would help the students to prepare and to be accepted in Graduate programs.
Brainstorming	This process was performed in the almost weekly meetings for project management. It was performed as part of the agile process, too.
Driving Question	Is a web system for managing academic projects based on project management concepts a valuable asset for improving the success and quality of dissertations and thesis?
Student Voice and choice	The students have helped to choose the tools and techniques for the system development at every phase.

Moreover, as an additional motivation it was clarified that the project was indeed a research project at the same time, which means that being part of the project would prepare them for entering a graduate program. Therefore, our motivation was to create a system that had a complete documentation, high source code comprehensibility, all aiming to increase the system's life cycle. Consequently, by themselves, the students realized that it was required a design refactoring, and a complete code refactoring. An

example of the use of code refactoring can be found in [14].

Phases 3 and 4 were natural consequences. From the beginning of Phase 2, we informed the students that the development team would have three levels of professionals: senior, full and beginner. Therefore, it was exposed the need to recruit and train new students for the project. From Phase 1 to Phase 2, the students Mariwaldo and Robson would be considered consultants and the two new students, Fernando and Glaubos were supposed to become the senior members of the team regarding the admittance of new students in the team. Some students tried to be part of the team, but they were informed that the treatment of the project was supposed to be professional, which meant a high degree of commitment, then only Valéria and Itamar remained as team members.

IV. MVC-EA-IRPM

We have employed for the web system development a set of methods and their combination. First, we used the Interdisciplinary Research Project Management (IRPM) [15], which is an approach for conducting interdisciplinary research of real problems using Project Management concepts [16] and problem-based learning [12], [17], second, we applied the Evolutionary Acquisition (EA) as system design methodology [18], and third, the Model-View-Controller (MVC) as the software architecture pattern [19], [20]. We start this section explaining IRPM followed by Evolutionary Acquisition resulting in EA-IRPM [7]. In the sequence, we briefly review the MVC pattern to present then the MVC EA-IRPM [8], [9].

In the Interdisciplinary Research Project Management (IRPM) - Figure 1, the Initiation phase starts with the determination of project goals, deliverables and process outputs, that is, the problem specification. Then to solve the problem, we identify two or more fields for an interdisciplinary approach, and then we have to document its constraints and assumptions, define strategy, identify performance criteria, determine resource requirements, define budget, and produce a formal documentation. The Planning phase consists of refining the project and of studying the problem based on the chosen fields. This study may help design or generate a new fundamental or methodology. Therefore, in the executing phase, an educational material may be prepared and applied in a classroom using a Problem-Based Learning approach [12], even if new concepts are not generated, a new system or technology may have been developed and then, it may be applied. If in Planning phase controls are established then educational, technological, economical and social parameters are available for measurement, allowing Control phase to be performed. In Closing phase the measures are analyzed. Finally, the Evolutionary Acquisition starts with the requirements analysis, as in Figure 2.

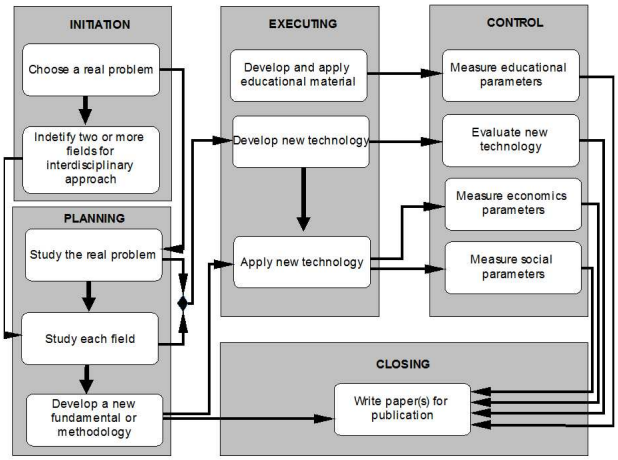

Figure 1. Interdisciplinary Research Project Management (IRPM), adapted from [15].

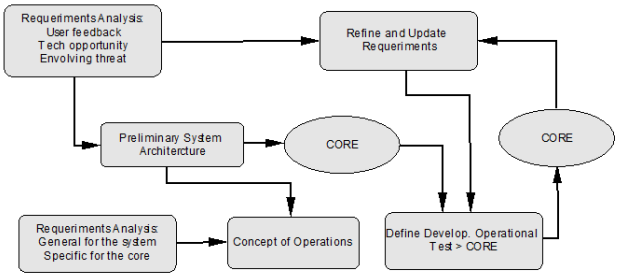

Figure 2. The Evolutionary Acquisition model, adapted from [18].

After defining the "general" requirements for the system and the "specific" requirements for the core, we elaborate the concept of operations. Then together with a requirements analysis of user feedback, technological opportunities and threats evaluation, we design the preliminary system architecture. From the system's architecture, we determine its core. New definitions and developments with operational test may generate a new version of the core. Then due to experience and use, we may identify requirements refinements inducing updates. It is worth mentioning that the EA separates the core of the system into blocks. If the system is a software, software engineering techniques may be applied.

The incorporation of EA into IRPM - called EA-RPM, consists of inserting EA into phases Planning, Executing and Control. In figures 3 and 4 RA means Requirements Analysis of: (1) general for the system and specific for the core; and (2) user feedback, technological opportunities and evolving threat. Hence, in Planning phase the attempt to develop a new fundamental or methodology consists of generating a preliminary system architecture beginning with RA 1, and then elaborating the concept of operations; and when available considering RA 2.

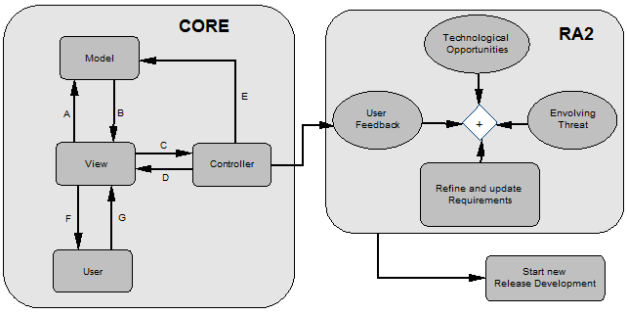

Figure 3. The Model-View-Controller Evolutionary Acquisition, adapted from [8], [9].

Executing phase consists of implementing the core from the preliminary system architecture followed by new definitions and developments of operational tests. Afterwards, we apply the system in a real life situation. Control phase is about refining and updating requirements, which implies in evaluating technology, measuring economic and social parameters, and verifying users' feedback, technological opportunities and evolving threats, that is, RA 2.

We use the Model-View-Controller (MVC). Consequently, it provides a way to split functionalities to independent development, testing and maintenance. We show the combination of MVC with EA in Figure 3. Thus, the core architecture of the web service is a modified MVC pattern connected to the Requirement Analysis (RA2), through the user feedback, which should be an independent database system. The letter links means: A - to query the model state; B - to notify view of change in model state; C - state view; D - user actions/commands; E - invoke methods in the models public APIs; F - output to user; G - input from user; H - to report problem/suggestion/requirement (psr). Connection between web services' core and Requirement Analysis occurs in the following way:

(1) User identifies a system's psr.
(2) User access the view to report psr, for instance, by pressing a specific button available in the user interface.
(3) View queries Controller state about psr.
(4) Controller notifies View of change to psr state.
(5) View displays psr state to user.
(6) User reports psr through View.
(7) View transmits user's report to Controller.
(8) Controller accesses users' feedback database system to report psr.
(9) Refinements and update requirements are defined using users' feedback, technological opportunities and evolving threats considerations.
(10) Decision to start a new release may be taken.

In this way, we may incorporate MVC into EA-IRPM, which we show in Figure 4.

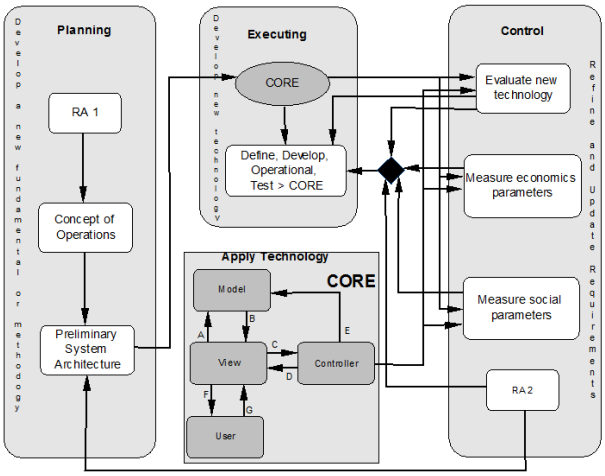

Figure 4. The MVC EA-IRPM Diagram, adapted from [8], [9].

The decision of releasing a new version of the core or a block is not automatic, and considerations of technological opportunities and evolving threats are still independent parts, that is, they are not necessarily part of the users' feedback database system, though they might be. Moreover, they are not automatic too. Consequently, error and fault detection mechanisms are important features to be considered as in [21]-[22].

Hence, MVC EA-IRPM presents evolution as an important factor in interdisciplinary web services systems. Actually, according to Breivold et al. in [23] "the ever-changing world makes evolvability a strong quality requirement for the majority of software architectures", i.e., MVC EA-IRPM aims to increase productivity and to facilitate software evolution.

V. Phase 1 – Prototype

The first phase of the system was developed during a one-semester class. It was a discipline of the Computer Science course, undergraduate level, called "Systems' Development", where we applied role-playing to simulate a software house.

We proposed to the students to develop a web system for managing academic projects. We explained to the students that the discipline would be conducted as a software house. That means that they would be divided in groups to develop a system where one professor (the professor of the discipline of "Systems' Development) would play the role of director of the company and the other professor (Patrick Letouze - the professor of the discipline of "Graduate Project") would play the role of the director of the client company. An additional student enrolled in the last discipline of "Graduation Project" would be the project manager (Robson), who was supervised in his graduation project by professor Patrick. Hence, it would be a project-based learning approach.

We informed the students that the web system was a real problem. That is, the system's aim was to provide a solution for the educational the discipline "Graduation Project" of the Computer Science undergraduate program of the Federal University of Tocantins. However, we commented that the system could support a Master Science or Doctorate program, too.

In the "Graduation Project", each student must develop a project, write a monograph about it and orally defend it to a designated committee or board that grades the student's work. That means basically, that we have a very similar structure to a Master Science dissertation or a Doctorate thesis. Actually, we told the students that in the new curriculum, the "Graduation Project" discipline would be split in two semesters and it would work similarly to a Graduate program. In this new scenario, we have a project proposal approval by a committee in the first semester, as a qualifying process (dissertation or thesis), and then the defense of the performed project (dissertation or thesis) to a committee in the second semester.

This work was published in 2011 [24], but in the fall of 2010, Ge *et al.* [25] published a similar experience, which the authors of this work were not aware of at the time of the conception and development of this academic project management system - the beginning of 2010.

The class started with 20 students (all male). One student quit the class due to professional reasons and another student failed due to excess of absence. The other 18 students were approved. Initially, the class was grouped into 3 teams according to individual aptitudes following the MVC 2 model (Model, View and Controller) [26].

The software implementation methodology was the Unified Process (UP). That motivated a second division of the class, because of UP phases [5] conception, elaboration, construction and transition. In the conception and elaboration phases all students participated focusing in their MVC team's responsibilities. Then in the construction and transition phases the students were grouped into 5 teams, 4 teams for the system's construction and 1 team for tests and integration. In the system's construction teams, there were at least 1 member of the interface team, 1 member of database team and 2 members of development team. In the test and integration team there was at least one member of each MVC team.

The chosen programming language was JAVA [27] and the framework was Javaserver Faces 2.0 [28], with some external components of Primefaces 2.2 [29]. The database management system was PostgreSQL 8.4 [30], with a relational database and the application server was Glassfish 3.0 [31]. For documentation and modelling it was used the Unified Modelling language (UML) [32], case use diagrams, class diagrams and sequence diagrams. Finally, the IDE was Netbeans [33].

As previously remarked, the system was not completely finished. At the end of the semester only the core of the system was completely functional. Hence, three students were responsible for completing the system: Robson - the development manager, Mariwaldo and Lesliê. After its completion, it was called SIGD 1.0 and only the student Mariwaldo continued working on the project in the next phase. The student Lesliê pursued other professional challenges, while the student Robson was hired as Information Technology manager in a Secretary of the State of Tocantins Governement due to his experience in this project, and a little after that he graduated. Professor Ary left the project to pursue his PhD degree.

The students enrolled in this semester reported that the experience of simulating a software house was profitable, especially because they felt the difficulties in decision making. They said that they had the feeling of a professional experience and that they had to deal with emotional and interpersonal issues, which were not required previously. Hence, in an overall sense they enjoyed the PBL approach.

VI. Phase 2 – Code Refactoring

Previously, we have developed an academic project management web system called SIGD 1.0 [24]. We conceived it based on project management concepts to serve as a tool for improving results of Graduation Project in the department of Computer Science at UFT. However, SIGD 1.0 had some productions problems such as unreliable code, small-scale development and adaptation issues for future changes. This was because of students' inexperience, who were not much acquainted with programming frameworks and agile development techniques. After all, our objective was to prototype the system, and most importantly, to provide students a real software development experience in a team environment as close to reality as possible. Consequently, in order to facilitate improvements and the system's evolution, we decided that a complete code refactoring should take place.

This second phase started with a new team. This time it was a project not related to any discipline, though the students used this experience to comply with the requirements of the "Supervised Internship". Professor Patrick was the supervisor and the students were Mariwaldo, Fernando, Glaubos and Valéria. This phase started with the first three students and only after some time Valéria joined the team. The first proposed task to the team was to study the release 1.0 of SIGD. The students were oriented to study the system with an open mind, but without any prejudice, as hoped, they realized by themselves that it was more interesting to start over the system's development. This approach intended to stimulate students' independence and maturity. At first, they were worried about proposing to perform a complete code refactoring because they were not informed that the first version was considered a prototype by the supervisor. Then after the evaluation meeting, where they independently

proposed to discard the first version. Therefore, the supervisor asked them to evaluate the tools and techniques to develop the system. They have acquired confidence to study the implementation strategies such as frameworks, database, tools and techniques, but the programming language JAVA and MVC EA-IRPM were induced.

Therefore, the students proposed to employ agile development techniques and tools such as Scrum, Hibernate, Spring Security framework, Pretty Faces and Maven. This new version - SIGD 2.0, was published in [6]. About the new version, in refactoring the older one completely by discarding all previous code, we confirmed that production time greatly reduces by applying agile development techniques and tools. An especial improvement was scalability providing a better basis for future developments.

The students decided to use Javaserver Faces as the standard for building user interface [28], Prime Faces as the source Javaserver Faces components [29], Glassfish as the application server [31], Netbeans IDE as the integrated development environment (IDE) [33]. Then differently from the first version, they decided to use:

- Scrum as a process of iterative and incremental development for an agile software development [34].
- Hibernate as an object-relational mapping (ORM) library for the Java language [35].
- Spring: security framework 3 as a framework for access control and authentication [36].
- Maven as a tool for management and automation of projects in Java [37].
- Prettyfaces as a filter-based servelets extension with support to JSF and to create URLs [38].
- MySQL as relational database manager [39].

There exist some major differences between release 1.0 and 2.0 of SIGD. In version 1.0 the system had, unrestricted file uploads, initial module of internationalization, and the user could have a class in which he was registered. In contrast, the second version included the project area, dissertation or thesis chapters, restricted and validated file upload. Additionally, hierarchical changes in the relationship of the objects optimizing search, automatic generation of projects and chapters, banking functionality, internationalization were implemented. Now a user had several classes avoiding the creation of multiple entries and a help manual. They accomplished to reduce response time of dynamic pages rendering them using Primefaces' AJAX.

In Table IV, we present a comparison between versions 1.0 and 2.0. The frameworks used in version 2.0, as hibernate, transform the classes into tables, which liberates the developer of manual labor, and in addition, it maintains the program portable to any database. Spring Security brings a solution for security needs. About Maven, it is a simple basic project structure to code compilation and to run unit tests, and to build the jar file extension. For last, the URLs has become "friendly" with Pretty Faces.

Table IV
COMPARING SIGD 1.0 AND SIGD 2.0

Tool	**SIGD 1.0**	**SIGD 2.0**
Software architecture	MVC	MVC
Programming Language	Java	Java
Server	Javaserver Faces	Javaserver Faces
Component suite	Prime Faces	Prime Faces
Application server	Glassfish	Glassfish
IDE	NetBeans	NetBeans
Development method	Unified Process	Scrum
Object-relational mapping	-	Hibernate
Application framework	-	Spring: security framework 3
Build automation tool	-	Maven
URL rewrite filter	-	Pretty Faces
Database	PostgreSQL	MySQL, PostgreSQL

Despite the time required for refactoring the system, the results compensated the loss of time, because the code was optimized and it had become ready for more updates and new developments. The use of more frameworks and efficient production methods made possible to create a more solid platform that serves as a basis to other applications, faster coding, and more comprehensible and clean programs with lower failure probability. Finally, in the new version, we had implemented it with MySQL to test the use of Hibernate, that is, it was enough to only change a few parameters to the system work with PostgreSQL, which have confirmed the database abstraction.

In this phase, the students reported that they felt that they had achieved a more professional maturity level, which indicates that a discipline of code refactoring could be used to obtain that for an average student. Indeed, they also said the fact that the project was about a real system to be used at the University was an encouraging factor. Additionally, the perception that they were involved in a research and the necessity to understand the structure of Graduate Programs were enough motivation to induce them to continue their studies in the graduate level.

VII. PHASE 3 – RELEASE TO MANUFACTURING

A new version of SIGD, version 2.1 was published in [10]. At this phase the purpose was to develop a more stable version ready to be released to manufacturing. Here the MVC EA-IRPM has become not only a strategy, but it has become the system itself.

Initially, the team remained the same as in phase 2. Then the student Itamar entered the team and after his entrance the student Mariwaldo has graduated. Mariwaldo at the end of his course prepared himself for a public selection for tenured

position (10 positions available for the entire state) of IT manager for the State of Tocantins Government and he has ranked number 1 among hundreds of candidates. Because of that, Mariwaldo had to leave the project.

In this new version the academic project management system was redesigned to have multiple instances that we call from now on SIGD_*i*, where *i* is an index number denoting different systems that may or may not be located geographically apart. The second part that we denote from now on SIGD-RA, which is a modification and reuse of SIGD that implements a database for the requirements analysis such as RA 2.

SIGD-RA is a web-based system that communicates with every registered SIGD_*i*. That means that many institution or Universities may have their own SIGD, and for requirements evolution, it communicates with only one SIGD-RA. For the users, it seems that they are using their own system, because communication is incorporated in the system's interface, that is, the MVC EA-IRPM model has only one View. Therefore, scalability is assured for the SIGD system.

SIGD_*i*'s interface has in its menu a "User Feedback" option that gives access to the SIGD-RA system. When accessing that option, the user may report a system's problem, suggestion or requirement. Additionally, it may be really a user's feedback, or a report of a technological opportunity, or an evolving threat identification. From the system's point of view in SIGD-RA, the requirements analysis is a project such as an academic project in a regular SIGD, that is, we have completely reused SIGD's core to develop a new application or web service. Besides, user's reports are managed through the system - SIGD-RA.

At the end of this phase, the students reported a feeling of realization because of the software registration. This process showed that the documentation elaboration process is very important and usually overlooked by developers. The requirements of registration demanded higher quality in the documents of the project, and this assessment provided a new perception about software development documentation. It is an evidence that a discipline at the end of the course could approach the software registration process in a PBL fashion in order to improve awareness of documenting development.

VIII. Phase 4 – Deployment

The system deployment consists of all of the activities that make a software system available for use. In our case, it means to make it available on web to the graduate programs of UFT. In fact, our client is the Dean of research and graduate studies.

First, before the deployment process, in this phase we submitted our source code for registration at INPI (Instituto Nacional de Propriedade Intelectual - National Institute of Intelectual Property), in other words, we asked for the patent of our software [11]. This task was perfomed by one of the students - Itamar.

Second, we have presented the system to the professionals of the directorship of graduate programs and to the directorship of research. It is worth noticing that professor Patrick is a former director of research of the University and Itamar did his supervised internship in this directorship, and that facilitated the communication and interaction with the professionals of both directorships. However, this interaction created a new demand of a special module for registration and control of research projects. Valéria and Itamar performed this new development.

Valéria and Itamar executed the deployment. It was a process of technology transfer to the Information Technology Directorship of UFT. They have provided all infrastructure for hosting the web system. They have become responsible for the system's maintenance, too. The system is going to be available officially to the graduate programs and for registration and control of research projects on the first day of January of 2016.

At this stage the students reported the weight of responsability about having the system in operation and subject to external criticizing. Practical issues have emerged about infrastructure, which most of the time, students tend to undervalue. Moreover, the perception of the necessity of specialists in different roles in software development has become evident.

IX. Discussion

We have developed a web system for managing academic projects using a Project-Based Learning approach to a real-life problem. The project initiated in the very beginning of 2010 and its deployment was concluded at the end of 2015. Our PBL approach was simultaneously curricular and extra-curricular. Because of 6 years of development, this approach was performed not in one semester nor within one class of students, it was performed with different groups of students in two different PBL settings. Hence, to the best of our knowledge, we have performed a PBL approach in an original way. The system's development occurred in a PBL fashion composed of four phases: prototype, code refactoring, release-to-manufacturing and deployment. In every phase the students had an active role in choosing tools and strategies, they received simple guideness and general requirements, it was their responsability to make choices, propose solutions and to defend them.

In addition to this pedagogical approach, in the phase 1 - prototyping, we simulated a software house with role-playing in a curricular discipline. In an extra-curricular mode, the students who were part of the development team of Phases 2, 3 and 4 in the project had the opportunity to be part of the system's recreation and evolution.

Breivold *et al.* in [23] observed the importance of software evolution in a systematic review. They concluded that:

> It is necessary to establish a theoretical foundation for software evolution research... to combine appropriate techniques to address the multifaceted perspectives of software evolvability.

If we observe that this work was the implementation of a web system using a methodology that combines well-established techniques for web systems evolution – the MVC EA-IRPM, then we will realize that the students who were part of the development team in Phases 2, 3 and 4 were not only exposed to the issue of software evolution, they actually learned a methodology designed for software evolution. They have learned that MVC EA-IRPM supports web services design, development, maintenance, and evolution. It is both a model and a method that takes advantage of the MVC architecture pattern for designing the web system, of the Evolutionary Acquisition for developing systems continuously using requirements analysis together with systems testing, and of project management concepts in an interdisciplinary research context through IRPM.

Regarding the students involved in Phase 1, we only have news about the three students who finished the first version of the system. All three are still working with software development. Robson and Mariwaldo wish to obtain a degree of Master of Science in Computer Science, but Mariwaldo has just achieved a tenured position as CSI and Robson has returned to the state of Tocantins to a tenured position in Information Technology at the City Hall of Palmas. Lesliê reported that he is going to apply to the next selection process of the Master Science program of Computational Modeling of UFT.

The other students who worked in the development teams of Phases 2, 3 and 4 are all enrolled in graduate programs. Glaubos has finished his Master Science in Computer Science at UFF (Universidade Federal Fluminense) and now is a PhD candidate at UFRJ (Universidade Federal do Rio de Janeiro) in the graduate program o systems engineering and computation. Fernando is finishing this year his Master Science at UFSCAR (Universidade Federal de São Carlos) in the graduate program of computer science and he has been invited to continue his studies as a PhD candidate in the same program. Valéria has started her Master Science at UFRJ in the graduate program o systems engineering and computation, and Itamar is finishing his Master Science at the Master Science program of Computational Modeling of UFT. They both report that they have the intention of doing a PhD.

Hence, it is remarkable that all students participants of the project demonstrated a genuine interest in pursuing their studies in graduate programs. In particular, the students participants in Phases 2, 3 and 4 have enrolled in graduate programs, which may be correlated to the authentic achievement of Phases 2, 3 and 4, in Table III. The same may be true for the three students who finished the system in Phase 1, in Table II. That is, they are professionals in IT and report their will in doing a Master Science related to computer science or Information Technology.

Hence, we have demonstrated that the proposed PBL approach was successful for developing a large and complex system for the University, though it took several years to accomplish that, which is probably a consequence of reduced development teams during Phases 2, 3 and 4. As future work, additional research concerning our PBL setting should be conducted to investigate the correlation between authentic achievement and the will to continuing studies.

Besides that, we are integrating our system into the computational platform of the Brazilian agency responsible for regulating all graduate programs in Brazil. The platform is called Sucupira and the name of the agency is CAPES, which is part of the Ministry of Education. We hope we have a successful integration with Sucupira and that our web system becomes available to all graduate programs in Brazil.

Finally, the user has become real and it is a fearful thing.

REFERENCES

[1] C. R. Rupakheti, S. Chenoweth, *Teaching Software Architecture to Undergraduate Students: An Experience Report*, 37th IEEE International Conference on Software Engineering, 2015.

[2] M. Jazayeri, *Combining Mastery Learning with Project-Based Learning in a First Programming Course: An Experience Report*, 37th IEEE International Conference on Software Engineering, 315-318, 2015.

[3] H. Topi, J. S. Valacich, R. T. Wright, K. M. Kaiser, J. F. Nunamaker Jr, J. C. Sipior, G.J. de Vreede, *Curriculum guidelines for undergraduate degree programs in information systems*, ACM/AIS Task Force, 2010.

[4] K. Gary, *Project-Based Learning*, Computer, vol 48, issue 9, 98–100, 2015.

[5] R. S. Pressman, *Software engineering: a practitioner's approach*, Palgrave Macmillan, 2005.

[6] F. Chagas, G. Climaco, M. G. Caetano, A. H. M. de Oliveira, P. Letouze, *Reporting a code refactoring and evolution of an academic project management web system*, 2011.

[7] P. Letouze. *Incorporating Evolutionary Acquisition into Interdisciplinary Research Project Management*, International Conference on Education and Management Innovation, Singapore, 2012.

[8] P. Letouze, M. G. Caetano, J. Y. Ishihara, D. Prata, G. Brito. *Applying MVC to evolutionary acquisition IRPM*, International Proceedings of Computer Science and Information Technology, vol 45, 123–128, 2012.

[9] P. Letouze, M. G. Caetano, J. Y. Ishihara, D. Prata, G. Brito. *Evolving Interdisciplinary Research with Model-View-Controller Evolutionary Acquisition Interdisciplinary Research Project Management*, Advanced Science Letters, 19, 8, 2170–2173, 2013.

[10] P. Letouze, M. G. Caetano, F. Chagas, G. Climaco, J. Y. Ishihara, *A Web Academic Project Manager based on MVC Evolutionary Acquisition IRPM*, International Proceedings of Economics Development and Research, 48, 179–187, 2012.

[11] Programa de Computador (*Computer Software*), *Propriedade Industrial*, Dados, 2326, 04/08, pag 080, 2015.

[12] J. R. Savery, *Overview of problem-based learning: Definitions and distinctions*, The Interdisciplinary Journal of Problem-based Learning, 9–20, 2006.

[13] W. N. Bender. *Project-based learning: Differentiating instruction for the 21st century*, Corwin Press, 2012.

[14] S. A. M. Rizvi, Z. Khanam, *A methodology for refactoring legacy code*, 3rd International Conference on Electronics Computer Technology, 6, 198–200, 2011.

[15] P. Letouze. *Interdisciplinary research project management*, International Proceedings of Economics Development and Research, 14, 338–342, 2011.

[16] K. Heldman. *PMP: Project Management Professional Exam Study Guide*, Sybex, John Wiley & Sons 2009.

[17] O. Pierrakos, A. Zilberberg, and R. Anderson. *Understanding undergraduate research experiences through the lens of problem-based learning: implications for curriculum translation*, Interdisciplinary Journal of Problem-based Learning, Purdue University Press, 4 (2), 2010.

[18] The Defense Acquisition University Press, *Systems Engineering Fundamentals, Supplement 2-B Evolutionary Acquisition Considerations*, Fort Belvoir, Virginia, Jan 2001.

[19] T. Reenskaug. *Thing-Model-View-Editor – An example from a planningsystem*, technical note, Xerox Parc, 1979.

[20] T. Reenskaug. *Models-Views-Controller*, Technical note, Xerox Parc, 1979.

[21] H. Xu, W. Chen, H. Qian. *Research on error feedback mechanism of information system*, 2nd International Conference on Artificial Intelligence Management Science and Electronic Commerce (AIMSEC), 5112–5115, 2011.

[22] H. Okamura, Y. Etani, T. Dohi. *Quantifying the effectiveness of testing efforts on software fault detection with a logit software reliability growth model*, Software Measurement, Joint Conference of the 21st Int'l Workshop on and 6th Int'l Conference on Software Process and Product Measurement, 62–68, 2011.

[23] H. P. Breivold, I. Crnkovic, M. Larsson. *A systematic review of software architecture evolution research*, Information and Software Technology, Elsevier, 54 (1), 16–40, 2012.

[24] P. Letouze, R. A. Ronzani, A. H. M. Oliveira, *An academic project management web system developed through a software house simulation in a classroom*, Proc. of 2011 International Conference on Sociality and Economics Development, 2011.

[25] X. Ge, K. Huang, Y. Dong, *An investigation of an open-source software development environment in a software engineering graduate course*, Interdisciplinary Journal of Problem-based Learning, Purdue University Press, 4, (2), 2010.

[26] Hibernate Faces, *EJB 3 Persistence e Ajax*, Rio de Janeiro: Editora Ciência Moderna, 2007.

[27] http://www.oracle.com/technetwork/java/javaee/overview/index.html

[28] J. Holmes, C. Schalk, *JavaServer faces: the complete reference*, Inc. McGraw-Hill, 2006.

[29] http://www.primefaces.org/

[30] http://www.postgresql.org/

[31] http://glassfish.java.net/

[32] http://www.omg.org/spec/UML/2.0/

[33] http://netbeans.org/index.html

[34] S. Ken, *Agile Project Management with Scrum*, Redmond, WA. Microsoft Press, 2004.

[35] W. Iverson, *Hibernate: A J2EE (TM) Developer's Guide*, Addison-Wesley Professional, 2004.

[36] P. Mularien, *Spring Security 3*, Packt Publishing Ltd, 2010.

[37] http://maven.apache.org/

[38] http://ocpsoft.org/prettyfaces/

[39] http://www.mysql.com/l