FINAL YEAR PROJECT MANAGEMENT PROCESS

Carlos López, David H. Martín, Andrés Bustillo and Raúl Marticorena
Área de Lenguajes y Sistemas Informáticos, Universidad de Burgos, Escuela Politécnica Superior
Edf. C 09006 Burgos, Spain

Keywords: Final Year Project, Process Specification, Product Metrics, Process Metrics.

Abstract: Development of final year projects on current IT degrees is one of the key subjects at the end of the studies. It is besides a key factor on degree and masters offers in European convergence process, and there is little discussion on the necessity to instruct students how to develop a project which can bring together the knowledge and competences acquired throughout their studies. In this work, a subject management model within a university context is proposed. Good Software Engineering Practices are applied: monitoring, process control, reviews, evaluation and measurement. Based on this model an automated-tool is described, along through two case studies showing results of its application. The first case study shows the products obtained out of the process' application. Second case study shows the results obtained when using the evaluation tool to compare final year projects in two different subjects: on one hand for the IT Engineering Hons (Ingeniería Informática II, 5 years) and, or the other hand, for the IT Management Engineering (Ingeniería Informática de Gestión ITIG, 3 years).

1 INTRODUCTION

There are some IT project development related subjects, usually named as "Final Year Project" (FYP), for current IT and future degree and master courses' curricula,. Regardless of the delivering University, the subject has some distinctive features: large number of professors involved, high amount of credits ECTS, individual tutoring, customized contents for each project, high amount of information to manage with, special institutional rules, framework of relations with companies, etc. In order to pay attention and improve these features, through improvement plans, a management process must be defined and a set of indicators, as well as process and product metrics.

Final year project management is a subject broadly discussed on Software Engineering, both from the industrial (Pressman 2005), (Sommerville 2006), (IEEE 2005), (Larman 2004), (Schwaber 2004) and educational point of view (IEEE 2004). Final year project management processes involve many activities: planning, configuration management, monitoring, control, reviews, evaluation and measurement (IEEE 2005). The final year project subject itself becomes a practical teaching framework to some activities, both for students and involved tutors (Fernández, Martínez et al. 2007).

In a university context there exist a large number of projects containing a great amount of associated documents' information to be processed, and little working time assigned to the involved tutors. The context of the university project development, from the point of view of techniques and tools used, is quite heterogeneous, making the automated collection of information nearly impossible. This heterogeneity is one of the main strengths of the subject, providing it with a high degree of innovation, both on tools and techniques. However, it is at the same time one of its weaknesses, in the sense that it makes difficult to handle the joint management of some of the industrial activities. Professors should not be discouraged by this situation when applying Good Software Engineering Practices, adapted to the available time. On the other hand, it may mislead students to have a distant view of the concept of project management both in productive enviroments, such as industrial ones, and in experimental projects, such as university context.

This study contributes to progress in the project management process, adding and adapting some typical Software Engineering activities such as: monitoring and process control, reviews, evaluation and measurement. It also underpins the importance

López C., H. Martin D., Bustillo A. and Marticorena R. (2010).
FINAL YEAR PROJECT MANAGEMENT PROCESS.
In *Proceedings of the 2nd International Conference on Computer Supported Education*, pages 5-12
DOI: 10.5220/0002768400050012

of encouraging, in a practical way, a quality policy among students.

The remainder of the paper has the following structure: Section 2 describes the main motivations of this work leading to process improvement, whose partial specification is shown in section 3. Section 4 describes an automated-process tool. Section 5 shows a case study, with all process tasks being applied. In Section 6 ,a project evaluation task has been tested, being the basis of an experiment where are compared Final Year Project subjects of the IT Engineering Hons (II, 5 years) and the IT Management Engineering (ITIG, 3 years) degrees. Finally, Section 7 lists some conclusions and future actions.

2 MOTIVATION

The large number of professors, both tutors and examiners at the examining board, requires a high level of management, not necessary to other subjects. An information base should be established to serve as a communication element between them.

As far as we know, current management processes do not keep experimental records of students' projects that would help on the FYP assessment and evaluation. As any other subject, it has to undergo the European adaptation process and ECTS credit assignation, where this information is required.

This work attempts to complement some works related to the management process tasks, and seeks for an automated approach in the university context. Specifically, (Dawson and Martín 2002) shows students' guides, (Polo Marquéz, Matínez Gil et al. 2007) introduces methodological aspects and evaluation tasks, and (Fernández, Martínez et al. 2007) tackles with relationships between tasks and roles. (Al-Shalabi, Chee et al. 2008) proposes an unconventional framework, incorporating new product development processes and practices, aiming to generate more balanced and productive graduates.

This work is intended to improve the quality of projects by incorporating measurement activities into the management process. This can assist to evaluate the process versus specific criteria, to identify its strengths, and record the conclusions. Therefore, every improvement can be analyzed as a change in the shape of an indicator or unit of measurement (ISO/IEC 2002). The incorporation of process metrics allows to progress in the search of a specific evaluation criterion, which can work as an internal self-assessment for students and as a knowledge base for tutors and examiners at the examining board. This new product measurement task improves educational methods through practical education, encouraging students to incorporate measurement processes in software development and quality culture.

3 PROCESS SPECIFICATION

This section shows the elements of a management process in Final Year Project subjects. For element descriptions, the applied notation follows the Software Process Engineering Metamodel Specification (SPEM), v. 2.0, (OMG 2006).

Section 3.1 provides an overview of the process. Lately, Section 3.2 it offers a thorough description of the project evaluation task. (López, Rodríguez et al. 2008) provides a more detailed description of the process elements.

3.1 Overview

Figure 1 shows the general structure of the process. The different relationships between the process elements are displayed: tasks, roles, and products. Showed relationships are: a role's responsibility towards a task and the obtained products as a result of a task execution.

Although there are rare variants on project management (Erasmus, internships, student proposals), the basic management process to Final Year Project Management is as follows:

1. Establishment of the basic rules for the subject management: examiners at the examining board, calendar, rules and literature.
2. A tutor offers his/her projects (title, description, number of students, publication date), which are sent to the responsible for the publication. The responsible for the publication makes adjustments to ensure that there are enough projects to fit the students demand, and publishes the project offers for the outgoing course.
3. Students discuss with the candidate tutors and then apply for preferred project. Tutor chooses applicant students. Student assignment to the project is notified to the responsible for the publication.
4. After the project has been assigned, the tutor is responsible for supervising. The several software products, user manual, installation manual, source code, etc. are developed.

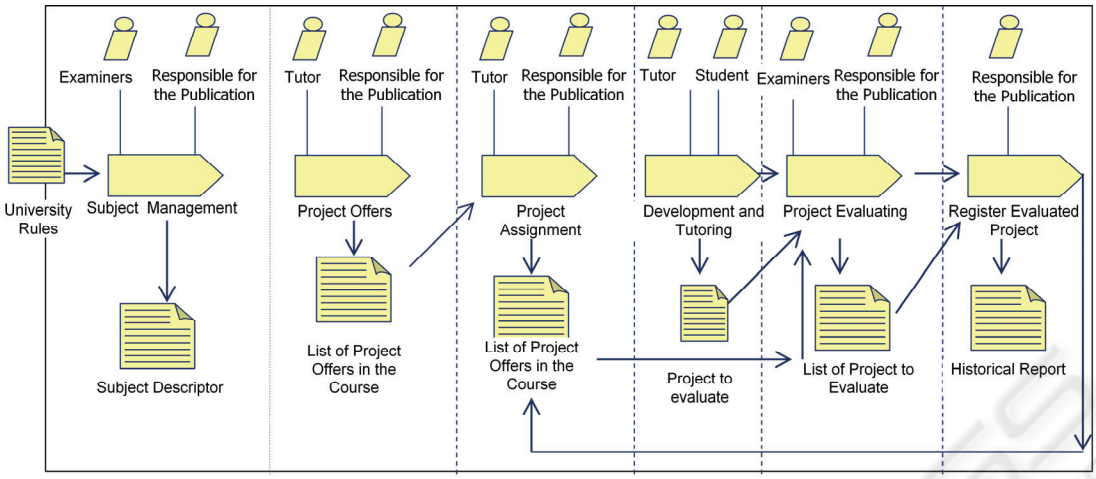

Figure 1: Final Year Project Management Process Specification.

5. The project is submitted to the examining board and examiners evaluate it.

6. At last, project data are registered, including an historical report on its development.

3.2 Task: Project Evaluation

Expired the project submissions period, examiners have a period of time to assess them. From the authors point of view, the establishment of a quantitative evaluating criterion, on which the final score relies on, with a thorough evaluating model, need a lot of time and adds excessive complexity. A mixed approach is proposed in terms of a qualitative evaluation of some features, combined with a quantitative evaluation of a specific software product. To face the time problem, data acquisition should be done through automated measurement tools. Generally, these tools provide optimal recommended values of their metrics. For each metric, two indicators should be taken into account: one to compare different projects from the same subject, and another one allowing comparing to external projects through recommended values. The internal projects' indicator defined by applying statistical measures: 25th percentile, median, 75th percentile, arithmetic mean and standard deviation, on all the projects.

Not all used metrics must have recommended intervals, for example size metrics. Furthermore, the metric set to be processed should stay as open as possible in order to allow the evaluation of new features in the projects. Restriction here is constraint to the time needed to collect the values and its objectivity. Once the metric set has been selected and its relative, all projects related, intervals calculated, it is possible to establish the potential improving needs. The beginning of the subject's improving process may be defined through the metric coverage indicators regarding recommended intervals. The number of metric values within the intervals is added and divided into the total number of values. Coverage indicator is equivalent to the concept of compliance defined on ISO product quality models (ISO/IEC 2001). Different types of coverage exist, depending on the statistical value used: coverage regarding arithmetic mean, coverage regarding median, coverage regarding the interval defined by 25th/75th percentile.

A specific project evaluation may be done from two approaches: using recommended intervals or intervals relative to other projects. Its individual coverage is calculated regarding the median. This allows establishing an objective criterion to evaluate and compare different projects.

4 FYP MANAGEMENT TOOL

One of the pursued goals on a process specification is to define some tools that could help to automate it. This specification should adapt to the facts allowed by technologies, tools and people.

One of the technological facts used by professors to manage their subjects' data are spreadsheets. Some of the features that stand them out from other data base systems include: easy handling on basic data maintenance operations and easy data transport and export. Another spreadsheet feature, not as popular, is its interoperability with other applications. In order to do so, information has to be structured through its name definition functionality.

The information to deal with is organized in four management levels, in such a way that a higher level includes the management of the immediately lower level. Levels correspond to the process generated products.

Level 1 generates the descriptor of the subject from the following information:

- Calendar (Description, Call, Date)
- Document (Description, Url)
- Rule (Description)
- Examining Board (Position, NameSurname, Nick)

Level 2 gets the level 1 product, and generates the current projects' list from the following information:

- Student (Number, NameSurname, Identity Card No, ReSits, Assigned)
- Project (Title, Description, Tutor1, Tutor2, Tutor3, Student1, Student2, Student3, Year_Assignment)

Level 3 gets the level 1, 2 products, and generates the submitted projects' list from the following information:

- Historical Report (Title, Description, Tutor1, Tutor2, Tutor3, Student1, Student2, Student3, Score, TotalDays, AssignmentDate, SubmissionDate)

Level 4 gets the level 1, 2, and 3 products, and generates the submitted projects evaluation' list from the following information:

- Experiment Description (Description)
- Metric Description (Description, ID, Type, MinValue, MaxValue, Visible)
- MetricValues (M0,M1,M2,M3,J0,J1,J2,J3,J4, J5, J6, J7, J8, J9, J10, J11, J12, J13, J14)

Out of the presented information, the one on metric description and metric values is to be highlighted. Each of the identifiers (ID) of the rows on the *MetricDescription* table correspond to the columns on the *MetricValues* table. In this way, the availability of a mechanism enabling the use of an open metric set is pursued. Furthermore, the table *MetricDescription* has a column entitled *type* to allow for different metric scales, either nominal or numerical. By doing so, the above mentioned goal on project evaluation task is achieved: the choice of the metric set used in the evaluation should be open to the decisions taken by the subject's responsible and its particular contextualization.

Structured data processing has been automated through a tool developed within the FYP subject itself. The tool is a java-developed information manager accessing the data with ODBC interface API. It generates the process specified products, such as HTML reports, whose content allows administrators to take well-grounded decisions.

Besides the automatic data processing, other functionalities have been incorporated on the generated reports:

- Stamp time
- Content licences
- Links to webpage lexicon validation
- Monitoring using Google Analytics system
- Content webcast and webcast monitoring through systems such as Addthis.

5 CASE STUDY: TOTAL MANAGEMENT

The process exposed thereon has been applied to the FYP subject IT Systems, taken on the 5th year of the IT Engineering Hons degree (II) at the University of Burgos, and the involved staff accessed the various product results through the teaching platform UBUCampus-e. This is an e-learning platform that provides resources for every subject: forum, assigned staff, student access statistics, e-community to create social networks. Access to the process results can be obtained at the following address http://pisuerga.inf.ubu.es/lsi/Asignaturas/SI/index.html.

The chosen metric set has been the one related to the source code, most of the projects have an associated code source. The measuring tool used was SourceMonitor (Campwood and Software 2007). The main criterion to choose this tool was its ability to calculate metrics on source code written in several popular programming languages (C++, C, C#, VB.NET, Java, Delphi and Visual Basic 6). Given the heterogeneous nature of the projects, this is considered as an essential requirement.

Some of the data corresponding to the various products obtained by applying the process to this particular context are described below.

The current project list contains the assignment information. Furthermore, it provides quantified data used to distribute the workload among tutors, and data to manage the assignment activity. There are currently 19 active projects, 2 of which are not assigned, and there are 15 involved tutors.

The submitted project list keeps a historical report containing submitted project data along with its quantitative information organized by course: number of offered projects, number of submitted projects, number of assigned students, number of assigned tutors, arithmetic mean of scores, arithmetic mean of development time in months (See

http://pisuerga.inf.ubu.es/lsi/Asignaturas/SI/HistoricoSist.html).

Furthermore, it records the statistical data (mean, minimum, maximum, standard deviation) related to the score (8.47, 5.3, 9.8, 0.95), and the data related to the time, in days, from project assignment to project submission (310.83, 120, 1.096, 212.7). This last effort indicator does not measure the real activity, but it can serve as a reference to estimate ECTS credits and evaluate potential improving actions to be taken.

From the evaluated projects' list, a code metric's set has been selected, as shown in Table 1. In every metrics' description, the assigned potential values' type (column Type) and, in the case of some numeric metrics, the intervals of tool-recommended values (columns MinValue and MaxValue).

Table 1: Metric Description.

Description	Identifier	Type	MinValue	MaxValue
Project's Name	M0	string		
Programming Languages	M1	string		
Lines of Code	J0	number		
Number of Sentences	J1	number		
% of Conditional Sentences	J2	number		
Nº of Method Invocation	J3	number		
% of Comment Lines	J4	number	8	20
Nº of Classes and Interfaces	J5	number		
Nº Methods per Class	J6	number	4	16
Avg Commands per Method	J7	number	6	12
Max Complexity	J10	number	2	8
Max Block Depth	J12	number	3	7
Avg Block Depth	J13	number	1	3,2
Max Complexity	J14	number	2	4

With metric set's values, a comparative measure is defined between the subject projects' statistical values (interval Q1=25th percentile and Q3 = 75th percentile, Avg = arithmetic mean, Med = mean) and the recommended values. This comparison is collected with the coverage's indicators showing the percentage of the values within the intervals (See Figure 2). Taking into account that two of the proposed measures, J10 and J12, evaluate maximum values, and the total of the considered measures is 7, it seems correct to keep a coverage regarding the 42.86% of the arithmetic mean. These coverage indicators may serve to establish global quality indexes of the subject.

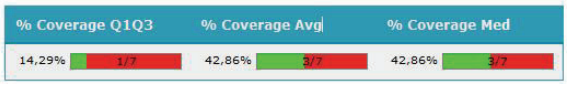

Figure 2: Coverage regarding recommended intervals in II.

The evaluation of each project is based on the measures collected for each of them and the comparison to two intervals: the ones recommended on (Campwood and Software 2007) (Table 1 Columns MinValue, MaxValue and the corresponding interval composed by percentiles 25th and 75th of the remaining subject projects'. With these comparison values, new coverage indicator is redefined serving as a comparison criterion (Coverage UBU). Figure 3 shows a project set evaluation (rows in the table) using these defined indicators. Moreover, it also shows the arranged values regarding the recommended intervals coverage (column Coverage Tool).

Project evaluation data of the evaluated projects' list do not allow a direct traceability over the submitted projects' list data, preserving somehow the privacy of these evaluations.

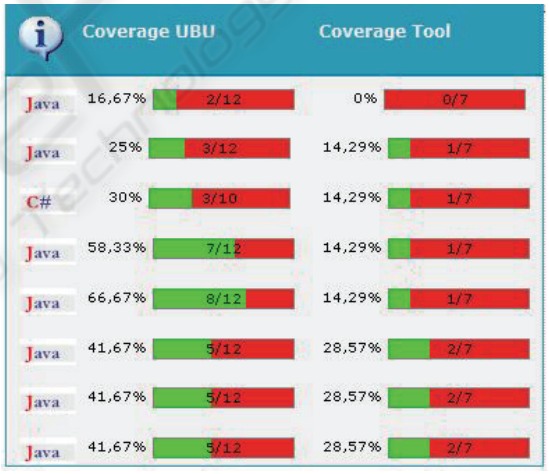

Figure 3: Project evaluations on II.

6 CASE STUDY: COMPARISONS IN DIFFERENT SUBJECTS

Once the project context has been established, the metric set selected, it is possible to establish global comparisons between subjects. On the selected context, the authors raise the following hypothesis: with respect to code metrics, are there differences between the projects submitted by IT Engineering Hons (II, 5 years) and the IT Management Engineering (ITIG, 3 years) degrees?

The project evaluation task exposed in the process (See section 3.2), has not been carried out in the IT Management Engineering (ITIG) degree. To simulate it, six professors from the IT department were asked for copies of projects from the last call submissions. Table 2 and Table 3 show the abstracts of the various experiments' sizes submitted in both degrees. Basic size measure is the analyzed number of lines of code (NLOC), and its breakdown with the various programming languages taken into account.

Table 2: ITIG Experiment Size.

	Total	Java	CSharp
NLOC	276.072	251.893	24.179
%		91.24%	8.76%

Table 3: II Experiment Size.

	Total	Java	CSharp	C
NLOC	876.806	779.164	82.806	14.836
%		88.86%	9.44%	1.69%

After statistical analysis (Q1=25th percentile and Q3 = 75th percentile, Avg = arithmetic mean, Med = median), for each metric, the coverage regarding the measuring tool's recommended values are calculated. Figure 4 shows the various coverage values obtained for ITIG.

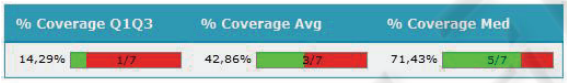

Figure 4: Coverage regarding recommended intervals ITIG.

When comparing the results with the II degree data (See Figure 2) it can be observed that same coverage values are maintained regarding the interval Q1 Q3 (14.29%) and coverage regarding Avg (42.86%). However there is a substantial improvement in the coverage regarding the median, since it goes from 42.86% in II to 71.42% in ITIG.

In the project evaluation it is important to mention that the minimum coverage regarding the tool values is 0% in II versus 28.57% in ITIG. This detail could be important to justify the improvement regarding the mean in ITIG. The evaluating process in II has been completed from 2003-2004 academic year, whereas in ITIG, projects were asked from the tutors, who later on stated that they offered highly rated projects.

Table 4 shows a comparison of two project features between both degrees; size and complexity, whose associated metrics have no recommended values in the measuring tool. The following metrics have been associated to the size feature: lines of code (J0), number of method invocation (J3), and number of classes and interfaces (J5). The following metric has been associated to the complexity feature: percentage of conditional commands. From these data, it is deduced that projects' size is pretty similar in both degrees. Taking the complexity associated metric' lower limits' values (J2), it may be concluded there is a higher complexity in II projects with regard to ITIG ones.

Table 4: Comparison between II and ITIG with metrics without recommended intervals.

		[Q1-Q3] II	[Q1-Q3] ITIG
Size	J0	[8.320 - 28.185]	[8.670 - 35.896]
	J3	[3.775 - 10.056]	[2.796 - 9.770]
	J5	[40 - 186]	[31 - 191]
Complexity	J2	[9.8 - 15.9]	[6 - 15.6]

Out of this comparison, it is concluded that, concerning code metrics, there is little difference between the projects submitted by students from II and the ITIG degrees.

7 CONCLUSIONS AND FUTURE WORK

A structure for a Final Year Project information management database has been provided, Good Software Engineering Practices being applied. Specially, a product and process metric set that could help both on subject global evaluation and on individual projects' evaluation has been defined. These have been adapted to a university context where some specific restrictions exist: professors and students' dedicated time, professors' workload, heterogeneous nature of the projects and public evaluation stage.

Actually, the products of the process are being used in IT Systems subject on the 5th year of the IT Engineering Hons degree (II). Figure 5 shows a summary of access data's obtained with Google analytics tool. We can observe two high visit peaks in september-october. The students must choose one project and request its assignment; therefore the activity is high during these months. The students also visit the web page to check the indicators, during the examination dates (february-june). This allows the comparison through the current level of coverage, using metrics.

Although the authors are aware of the limitations of the exposed evaluating criterion, out of these

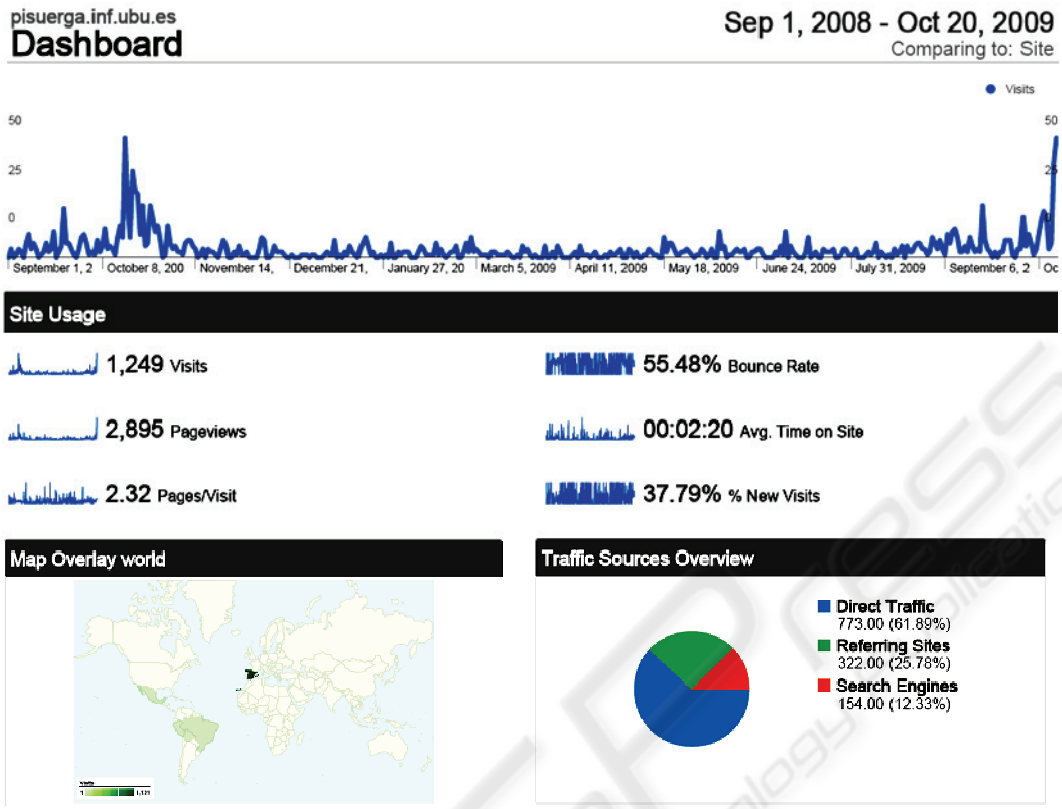

Figure 5: Google analytics report.

statistical measures of source code, it is proposed an objective way to evaluate the product obtained after carrying out of the project. Case study shown in section 5 provides a quantified reference to other projects that could serve as a base both for monitoring and comparison. Furthermore, comparison of different subjects is enabled, with a global approach.

Self-assessment habits of software development through product metrics have been improved. This way, it is possible to satisfy two of the knowledge units proposed at ACM SE 2004 (IEEE 2004), difficult to achieve in other subjects' context: software quality culture (2 hours) and activities related to project control (2 hours). This feature showed up with the incorporation of evaluations through metrics by the students in the technical documentation of their projects.

The management model proposed here may help to advance in the European convergence process in Final Year Project subjects. It defends educational methodologies where students get a more practical and personalized learning.

Facing the future, the defined process is open to the potential incorporation of new metrics coming from other information sources. The value acquisition is restricted to be done in an automatic and fast (< 10 minutes per project) way. In this sense, version control systems may offer great information on the real activity of the development process realized by the students.

The planed experiment of II and ITIG projects' comparison revealed that there are no main differences between them. To reinforce these results, it could be interesting to make a simulation with data from other Universities. Furthermore, it could be possible to answer new hypothesis: Are projects' size and complexity of different Universities comparable?

REFERENCES

Al-Shalabi, A., S. B. Chee, et al. (2008). *Framework for Orienting Engineering Undergraduate Final Year Projects towards New Product Innovation Process.* Norristown, Int Business Information Management Assoc-Ibima.

Campwood and Software, 2007. *SourceMonitor*. Available at http://www.campwoodsw.com/sourcemonitor.html

Dawson, C. W. and G. Martín, 2002. *El Proyecto Fin de Carrera en Ingeniería Informática: una guía para el estudiante*. Prentice Hall.

Fernández, J., A. B. Martínez, et al., 2007. *PFC: Los dos lados del espejo. Proyectista - Director ¿Una visión compartida?* VII Jornadas de Enseñanza universitaria de la Informática (Jenui 2007).

IEEE, C. S. , 2005. Guide to the Software Engineering Body of Knowledge: 2004 Edition - SWEBOK.

IEEE, C. S. A. f. C. M., 2004. *Software Engineering 2004 Curriculum Guidelines for Undergraduate. Degree Programs in Software Engineering*, National Science Foundation under Grant No. 0003263. A Volume of the Computing Curricula Series.

ISO/IEC, 2001. *Software engineering -- Product quality Part 1: Quality model*.

ISO/IEC, 2002. ISO/IEC 15939:2002, *Software Engineering-Software Measurement Process.*

Larman, C. (2004). *Agile and Iterative Development: A Manager's Guide*, Addison-Wesley Professional

López, C., J. J. Rodríguez, et al., 2008. *Gestión de Trabajos Fin de Carrera.* X Simposio Internacional de Informática Educativa (SIIE 2008).

OMG, 2006. *Software Process Engineering Meta-Model 2.0*.

Polo Marquéz, A., J. Matínez Gil, et al., 2007. *Hacia una metodología para el desarrollo de trabajos y Proyectos Fin de Carrera en Ingeniería Informática*. VII Jornadas de Enseñanza universitaria de la Informática (Jenui 2007).

Pressman, R. S., 2005. *Software Engineering: A Practitioner's Approach, 6/e*, McGraw-Hill Interamericana.

Schwaber, K. (2004). *Agile Project Management with Scrum,* Microsoft Press.

Sommerville, I. , 2006 . *Software Engineering, 8/e*, Pearson Educación.