# Tool for Validation Software Projects in Programming Labs

http://dx.doi.org/10.3991/ijep.v2i2.2080

A.J. Sierra, T. Ariza, F.J. Fernández, G. Madinabeitia
Universidad de Sevilla, Sevilla, Spain

***Abstract*—This work shows a testing tool used in Fundamentals of Programming II laboratory in Telecommunication Technologies Engineering Degree at University of Sevilla to check the student project. This tool allows students to test the proper operation of their project in autonomous way. This is a flexible and useful tool for testing the project because the tool identifies when the student has carried out a project that meet the given specifications of the project. This implies a high rate of success when the student delivers its project.**

***Index Terms*—Tool, Telecommunication Technologies Engineering, Fundamentals of Programming, Student-centered Learning System, First-Year Students, Statistical Analysis**

## I. INTRODUCTION

In this paper, we present Tool for Validation Software Projects (TSVP), a tool for testing the course Project, used in the Fundamentals of Programming II laboratory in Telecommunication Technologies Engineering Degree at University of Sevilla. This tool allows autonomous verification of course project by the student. We show a flexible and useful tool for checking the course project. Furthermore, it is used to identify when the course project fits the requirements. It has led to an increase in success rate in the evaluation of the student project in FPII.

The European Higher Education Area (EHEA) was launched along with the Bologna Process' decade anniversary, in March 2010, during the Budapest-Vienna Ministerial Conference [2]. The foundation of EHEA is based on student's work. Laboratories are essential to develop theoretical content and implement abilities in a larger scale work. The TVSP facilitates both the teacher's and student's work, in the context of a project-based learning methodology.

There are tools that automate completely or partially the task of assessment software (Computer Aid Assessment or CAA) at university scope [1][7]. Among these systems we highlight the following, BOSS [3], Curator [6], Goodle GMS [14], CourseMarker [11], Assyst [10], HoGG [15] and OnlineJudge [5].

Conversely to representation of assessment content and results [12] that is standardized, CAA has not been standardized.

Therefore, there is a problem of portability and interoperability between them, which makes it difficult to adapt these systems to other university environments.

This paper presents a tool for specific educational requirements in Fundamentals of Programming II in EHEA. The Tool for Validation Software Projects (TVSP) is implemented in C language. It has been based on previous implementations, in other programming languages. These languages are Perl [17] and shell script [16]. The objective of this work is to provide a simple tool for students. This tool is easy to use and is integrated into the working environment for the laboratory. This tool is implemented without an external server and can perform the tests without Internet connection.

This paper is organized as follows: first, the laboratory is shown in the context of the course Fundamentals of Programming II in Telecommunication Technologies Engineering Degree offered at the University of Seville in 2010-11. Next, we show the laboratory in which we use the tools. Then, we describe the implemented tool describing the functionality, goals and usability. And finally results and conclusions are shown.

## II. BACKGROUND

### A. EHEA

EHEA is a set of agreements adopted by all European countries to harmonize their university education. It includes not only the member states of the European Union, but a total of 46 countries.

The principles which regulate the EHEA are four:

a) A unit of measurement common training.

b) A common training structure.

c) Quality assurance.

d) Transparency of information.

The basic objective is to facilitate mobility of students, faculty and graduates among all member countries of the EHEA, so that students can continue their studies in another university system, it can generate faculty exchange programs and facilitates mobility for workers with higher education. Another aim is to make the EHEA an attractive environment to come to Europe to students from third countries, with the advantages it brings to the cooperation and international solidarity.

### B. The European Credit: A unit of measurement studies

Until now, University studies in Spain were computed using LRU [13] credits. These credits indicate only the number of class hours composing a curriculum (10 hours of class were 1 credit). The European Credit is a harmonization of the duration of the studies. All Courses have the same length, regardless of the degree you are pursuing. This allows an accurate assessment of training achieved at a given time if students want to or are obliged to change the University and facilitates the recognition of studies.

A European credit represents 25 hours of study and takes into account not only the session in classroom (including theoretical sessions, practical laboratory sessions, seminars or workshops), but also computes the hours of study to solve the various tasks assigned (exams, papers or conducting seminars).

Each course consists of 60 European credits. The EHEA does not change the time devoted to study, but makes it much more explicit and transparent.

### C. Changes in the University

The set of EHEA is structured in three levels:

- The basic level is the *Degree*. In Spain degrees consist of 240 European credits (4 courses). The degree enables to develop a profession that requires a university educational level.
- The second level is the *Master*. This is a specialization of a specific field of knowledge or multidisciplinary studies (representing disciplines from various areas). Guidance can be for research, for professional practice, or mixed. The duration of the Master will be ECTS 60, 90 or 120 credits (1 and 2 courses), depending on the nature of the studies.
- The third level is the PhD. In general, the PhD is a research process aimed at developing the thesis. To enroll in a doctoral program, the person concerned will have to be overcome at least 60 credits European Masters level or be in possession of a graduate degree of at least 300 European credits.

### D. EHEA at University of Sevilla

The University of Seville currently offers 77 degrees and double degrees and 81 Master Degrees [19].

### E. School Of Engineering at Sevilla

The implementation of the EHEA at the School of Engineering at Seville (Escuela Técnica Superior de Ingeniería, or ETSI) has been based on Aeronautical, Industrial, Telecommunications, and Chemical Engineering. These qualifications have enabled currently offered in the ETSI Grade the following qualifications:

- Degree in Aerospace Engineering
- Degree in Industrial Technology
- Degree in Telecommunications Engineering Technology
- Degree in Chemical Engineering
- Degree in Civil Engineering
- Degree in Electronic Engineering, Robotics and Mechatronics
- Degree in Energy Engineering
- Degree in Industrial Engineering

The postgraduate program is structured around seven Master's programs, each of them associated to a doctoral program:

- Master in Electronics, Signal Processing and Communications.
- Master in Electric Power Systems.
- Master in Automation, Robotics and Telematics.
- Master in Advanced Mechanical Engineering Design.
- Master in Industrial Organization and Management.
- Master in Environmental Engineering.
- Master in Thermal Energy Systems.

### F. Telecommunication Technologies Engineering Degree

In Telecommunication Technologies Engineering Degree (or TTED) at University of Sevilla, the techniques and technologies of transmission, treatment and information management, are studied. This will provide extensive knowledge of communications, electronics, signal processing, information management, and computer networks.

TTED offers four tracks or specializations:

- Telecommunication Systems
- Electronic Systems
- Telematics
- Sound and Image

Students accessing this qualification must have disposition to work and have analytical skills. You need a good background in mathematics and physics, and having language and information technology skills is highly recommended.

### G. First-year at TTED

The first-year for Telecommunication Technologies Engineering Degree at University of Sevilla consists of two terms.

Each term has different courses. The First term has the following courses:

- Mathematics I.
- Mathematics II.
- Physics.
- Computer Basics.
- Fundamentals of Programming I.
- The Second term has the following courses:
- Fundamentals of Programming II.
- Circuit Theory.
- Mathematics III.
- Statistics.
- Technology devices and Components.

## III. LABORATORY CONTEXT

This laboratory is taught in the subject of Fundamentals of Programming II (FPII) in Telecommunication Technologies Engineering Degree at University of Sevilla. In this section we show the context of the laboratory. To accomplish this purpose, we present the objectives, the contents, the used methodology and the evaluation system.

### A. Fundamentals of Programming II in EHEA

FPII in EHEA is a term core subject at the first-year of Telecommunication Technologies Engineering Degree. This subject consists of 6 ECTS credits corresponding to 150 hours of student work. 1.5 ECTS credits are dedicated to theoretical contents (15 classroom hours and 22.5 hours of study) and 4.5 ECTS credits are dedicated to practical classes (45 classroom hours and 67.5 hours of study). This subject has a high practical content.

### B. Objectives in FPII

The subject aims to consolidate basic skills obtained in FPI by solving a complex problem. The principles and fundamentals of programming are developed by means of problem's resolution, in practical sessions.

The objectives of this subject are the following:

- Resolution of problems through structured programming techniques by the decomposition of complex problems.
- Management and implementation of abstract data types.
- Development of programs in a programming language such as C, providing the methods for handling data structures.
- Design applications using Object-oriented programming (OOP) techniques. This technique includes features such as data abstraction, encapsulation, messaging, modularity, polymorphism, and inheritance in Java.

### C. Contents in FPII

The content in FPII is the following:

- Resolution of problems using abstract data types.
- Development of programs in a programming language such as C using abstract data types.
- Resolution of problems through OOP techniques. This technique includes features such as data abstraction, encapsulation, messaging, modularity, polymorphism, and inheritance in Java.
- Development and implementation of an application in an OOP language such as Java, using OOP techniques.

The practices carried out in the part corresponding to the abstract data types are the following:

- Basic operations on linked lists.
- Resolution of two problems using a linked list: One of them consists of reading words and counting the number of times that each word is repeated, and the other one consists of reading data from a file, inserting orderly them into a list and writing these data into a file.
- Sum of a certain number of polynomials using linked lists.
- Stacks and queues implemented with tables and lists.
- Circular lists.
- Use of trees.

The practices carried out in the part corresponding to OOP are the following:

- Programming environment tools (java, javac, jar, javadoc, jdb).
- Arrays.
- The Java Debugger (jdb).
- Handling Exceptions in Java.
- Encapsulation (classes, inheritance and packages) and utilities (String class and other basic classes).
- Polymorphism (overloading methods and constructors).
- Inheritance, overwriting methods.

### D. Methodology in FPII

The methodology used in FPII for the acquisition of knowledge is the following:

- Lectures: classes where the teacher makes a theoretical exposition of the subject matter, explaining the basics of the subject.
- Laboratory practices: these classes are dedicated primarily to the student address and resolve the problems proposed by the teacher.
- Active use of e-learning technologies: this technology has been used for storing contents, submission of practices, continuous evaluations, forums, internal e-mail. The virtual learning environment system used is WebCT (Blackboard Learning System) [22].
- Project-based learning, or PBL, is the use of in-depth and rigorous classroom projects to facilitate learning and assess student competence. Students use technology and inquiry to respond to a complex issue, problem or challenge. PBL focuses on student-centered inquiry and group learning with the teacher acting as a facilitator. Every student must complete two course projects that will consist of developing an application, comprising the steps of understanding the problem, designing of the program, coding and subsequent testing. One course project is in C, and other project in Java.

### E. Evaluation in FPII

The evaluation of the course in FPII is based in the following concepts:

- Attendance at practices: it is indispensable to attend a minimum number of practical classes, and the submission of some exercises developed during the practice sessions.
- Theoretical grade: tests and final exam is the 65% of the final grade.
- Tests: there are two tests with a value of 10% of the final grade. The virtual learning environment system is used in these evaluations.
- Final exam: the exam with a value of 80% of the final grade. It may contain a mixture of multiple choice questions (theoretical or practical issues) and problems to be solved by the student.
- Practical grade: Design and implementation of two course projects in C and Java is the 35% of the final grade.
- Course project in C: Development and implementation of programs in a programming language such as C for handling abstract data types. This project consists of a complete program whose requirements will be provided at the beginning of the course. This work must be defended by the student in an exam. The teacher will request new modification of the program.
- Course project in Java: Design and implementation of an application using Object-oriented programming (OOP) techniques. This project consists of a complete program whose requirements will be provided at the beginning of the course. This work must be defended by the student in an exam. The teacher will request new modification of the program.

So, we are using continuous evaluation in the subject. In addition to monitor what students are doing in the laboratory, there are two small tests before the final exam. Exams are distributed throughout the course and meet several objectives:

- Ensuring that the student has the skills to make the practices and understand the following topics of the course. It leads the student to organize the study of the subject properly. Furthermore, they must study a part of the subject before starting the next, so that classes and laboratory sessions are more useful.
- Training students in examinations of the subject.
- The student receives information on their current knowledge and he/she identifies gaps that must be overcome.

## IV. METHODOLOGY IN THE LABORATORY

The project-based learning is effective for students to acquire programming skills [4]. Projects must be evaluated by the teacher. The project-based learning has traditionally been used in courses with few students, but is difficult to implement in courses with many students.

Due to the large number of students in FPII, a tool is necessary to facilitate the evaluation of the student's course project. So, the teacher can validate easily and quickly the functionality of the completed project. Furthermore, this tool will also facilitate the development of the project by the student. The student can test the proper operation of the project in the development phase. The tool will report identified problems.

The course project works properly when it has passed a series of tests carried out using the tool. Then the student can delivers it to the teacher.

Access to the tool must be integrated with the learning environment used in the course. The virtual learning environment system used is WebCT. Therefore, students can get the tool through the virtual learning platform.

The student needs to have an environment similar to that used in laboratory practices. This development environment is used for the execution of the tool and the project. To this purpose, a virtual machine is provided to the student. This virtual machine is identical to the programming environment used in the practical classes. The virtual machine includes the same operating system and the tools necessary for compilation and execution of programs. We provide the necessary instructions to download the virtual machine.

The use of the tool must be easy and intuitive for the student. The tool keeps the environment (compilation and execution) used in the laboratory by providing a user friendly interface.

The review of the student project should be automated as much as possible to enable the correction of a large number of projects.

Students are examined of the project as follows:

- The student must deliver in advance the course project completed and tested with the tool so that it passes all the provided tests. The delivery takes place via WebCT.
- On the practical exam, the student must first download his project from the virtual learning platform.
- He will be required to perform new tests using the tool. In order to do this, we provide new input files to the program and the corresponding outputs.
- The student must perform certain program modifications that change some basic functionality. This ensures largely that the student is the author of the work and understands it perfectly. It also ensures that the student has organized the project in a suitable way and that the functions are intuitive (easy to understand).
- The student, before the end of the test, will use WebCT to deliver all the testing results. The student must make a copy of its modifications. This copy will be used if any problems arise, for example, if he sends the wrong files.
- Finally, the teacher verifies the result in the review itself or in the virtual learning platform. Check the program's structure, comments, legibility, etc ... to assign a rating.

## V. DESCRIPTION OF THE TOOL FOR VALIDATION SOFTWARE PROJECTS (TVSP)

### *A. Background*

There are tools that automate completely or partially the task of assessment software (Computer Aid Assessment or CAA) at university scope [1][7][3][6][14][11][10][15][5].

CAA tools help to assess more objectively. However, it requires more careful design of the tasks and the assessment criteria. It is necessary specifying strictly the results to be obtained and the test suite designed trying to cover all possible cases. These tests, along with the expected results can be provided to students so they can make tests before delivery of the work. A subset of the tests is provided to students. The students are warned that the given tests will only check part of the code, encouraging them to further testing, as done in [23].

Typically, a CAA tools use the model of black-box as a component of assessment.

The student program is considered a black box. CAA tools evaluate the program's output compared to a reference correct output for different input tests. Tests can be done to a whole program (to be compiled) or parts of a program (functions, code snippets, lines, etc.). In TVSP both options are possible. Although so far we have only assess complete programs.

In addition to comparing the output, often measurements of the efficiency of program are taken (CPU usage, execution time, memory usage).

The black box model requires running unknown code. It is desirable using a sand-box model (isolated and secure execution environment) in the implementation. So, it is prevented possible harm that may result from erroneous or malicious code when running dynamic tests.

In addition to the dynamic analysis of the program given by the student, CAA tools can perform a static analysis of the code delivered.

The static analysis can detect hidden problems that have not been obtained by the limited number of tests. The static analysis can detect hidden problems that have not been found by the limited number of tests

The compilation can be considered a basic static analysis of the code, and generates warning when it detects a possible error.

Sometimes, it is necessary to search for particular words or phrases in the code, for example to force or forbid the use of certain structures or functions.

However, as cited by reference [9], it is not possible to automatically evaluate all aspects of good programming. The automatic assessment should be supplemented with human evaluation. In this case, the tool may set the weight of the score given to each part of the evaluation.

This section shows the tool used at the first year in FPII. The Tool for Validation Software Projects (TVSP) allows the student a guided learning. In addition, the student can verify its course project in autonomous way[18]. The tool is implemented in C language. It has been based on previous implementations, in other programming languages. These languages are Perl and shell script.

Next, the working environment used for the development of laboratory practices is shown. Then, the targets set for the implementation of TVSP are described. After this, the architecture of the tool is shown and, finally how to use the tool.

### *B. Working environment*

The following environment was used for laboratory practices of FPII:

- Operating System: Ubuntu 11.04.
- Compiler: gcc and JDK.
- Debugger: gdb and ddd.
- Editors: emacs and gedit.
- Other tools: make.

A virtual machine is provided to students. Thus the project can be developed anywhere and everywhere.

The virtual machine provided is a clone of the operating system and applications used in the laboratory. This virtual machine is compatible with VMware Player [31] and VirtualBox [30] (freeware).

The advantages of using virtual machines include the following:

- Any student can work with the same tools from anywhere, regardless of operating system installed on its computer.
- Ensuring that the working environment exactly matches the lab environment used.
- Fault management.

The practical classes are held at the Computer Centre (CC) of the Engineering School. The CC currently has 12 classrooms with capacity for about 25 or 50 students. The PC's have the necessary environment for the different subjects taught at school.

In the current year (2011-2012) 5 lecture classrooms have been enabled so that students can bring their computers and do the practices.

The University of Seville provides each student a laptop with a low cost, refundable returning the laptop.

### *C. Objetives of the Tool*

The tool, in FPII, allows the student the following operations:

- Compilation the source code of the course project and check whether this operation is done without any errors or warnings.
- Tests of the memory management. This task verifies that all dynamic memory used by the course project is released.
- It runs a test suite and verifies that the output is properly generated.

This task includes verification that the output generated by the student's project matches the expected output. The verification tasks include the following points:

- Check the output files.
- Check the standard output.
- Check the standard error output.

### *D. TVSP Functionality Diagram*

The Student Project Code (SPC) is the starting point of the validation process.

A Global Error Report (GER) is generated at the end of this process. The Fig. 1 shows the TVSP functionality.

The validation has the following steps:

- Compilation of the SPC. This phase generates bug report, including compilation errors and warnings.
- Generation of Modified Student Project Code (MSPC) from the SPC. The MSPC includes the management of signals produced in exceptional situations. Furthermore, the MSPC tests the memory used by the program for each case.
- Compilation of the MSPC to generate the Modified Executable (ME).
- Execution of the program through the use of tests to generate the error output report. This error output report is included in the GER. In TVSP, the SPC is tested in an isolated virtual machine (no internet connection) similar to that used by students in laboratory practices.

The error report generated from the execution of the ME includes the following information:

- Standard Output Report (SOR). This shows the differences between the expected standard output and standard output obtained by the student’s project.
- Standard Error Output Report (SEOR). This shows the differences between the expected standard error output and standard error output obtained by the student’s project.
- Generated File Report (GFR). This shows the differences between the expected output files and output files obtained by the student’s project.
- Memory Report (MR). This shows the errors in memory management in the student’s project.

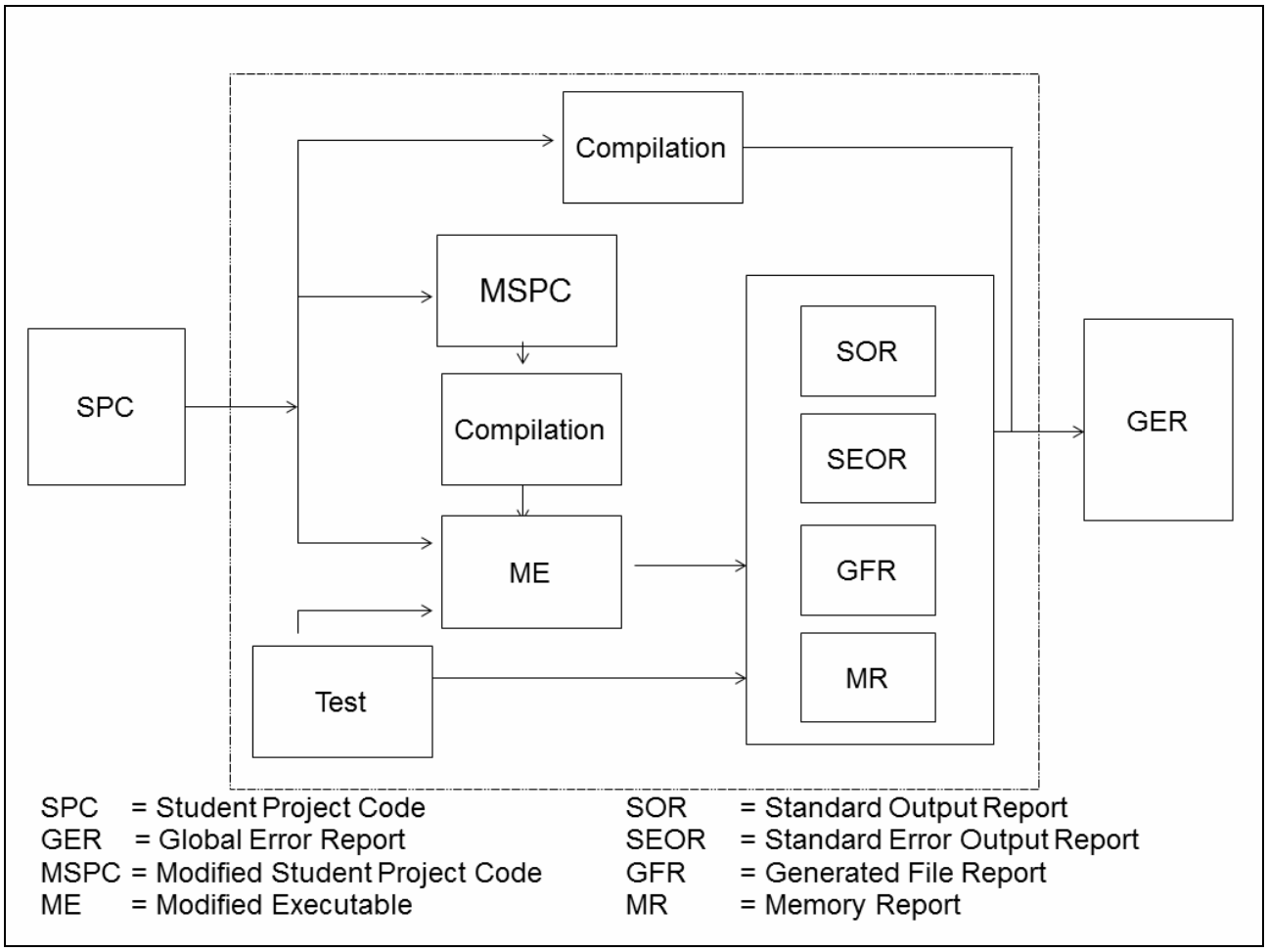


Figure 1. TVSP Functionality Diagram.

### *E. Using TVSP*

Both the tool and instructions to use are provided in the virtual learning platform WebCT.

The tool runs on the command line. The user's manual is provided in a file README.txt. The user interface of the tool is simple and intuitive for rapid deployment and efficient management by the student, as shown in Fig. 2.

In the course 2010/11 the course project consists of emulating a Network Address Translator (NAT). The program reads a NAT configuration file and an input file with the packets that are received to NAT. The program generates an output file containing the translated packets. The application uses linked lists to store the list of translations used by the NAT and to store the packets arriving to it.

TSVP has been used during the course for different purposes:

- Summative assessment: Summative assessment is that which is used to assess the student's final work. Retry is not allowed. Initially this was the main use of the tool.
- Formative assessment: In formative assessment is considered that the correction is mainly used as feedback to the student, for showing its failures and enabling a better implementation. In this case, the student can perform multiple deliveries. In our case this assessment is done prior to delivery, because the student has the assessment tool.
- Online exams: During the online exams student must use this tool. In the evaluation, students must make minor modifications to the code. Other authors have made similar actions [8].

### *F. The management system to deliver in TVSP*

Other tools include an assignment management system. In FPII, WebCT, as virtual learning platform, facilitates this task. WebCT allows us to have a collection of issues available to students (that can be performed by levels), generate reports and statistics, post results, etc.

## VI. RESULTS

TVSP has been used at the first year in FPII laboratory in Telecommunication Technologies Engineering Degree at University of Sevilla for establishment of EHEA.

This tool allows a guided learning of the student. In addition, the student can independently evaluate the proper functioning of the project. TVSP has been implemented in C and has been based on previous deployments in other programming languages (Perl and shell script).

The course project examinations are realized on various dates, spread along the term. TVSP allows students great flexibility for the evaluation of their work and to organize their efforts.

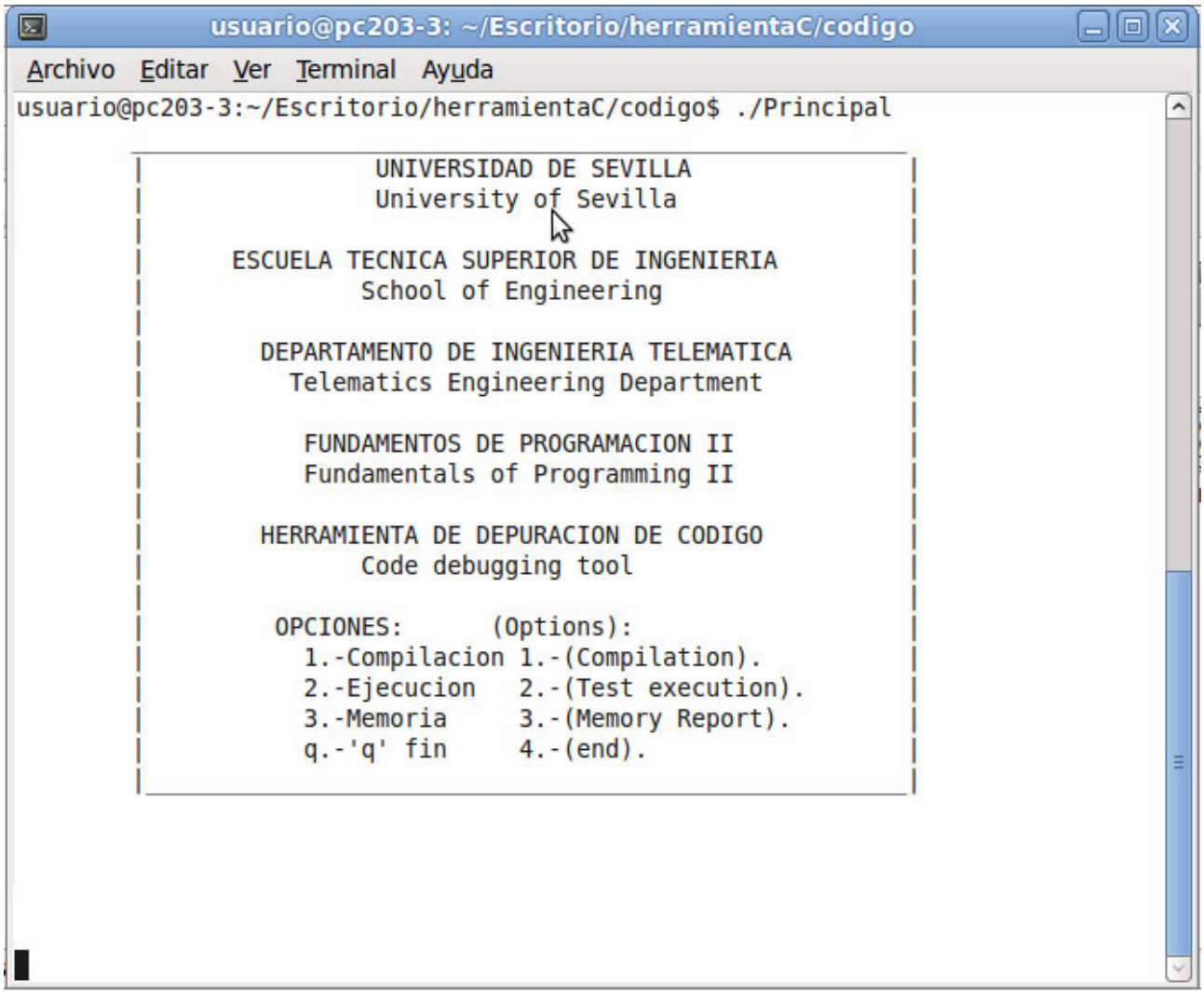


Figure 2. User interface of the tool.

Figure 3 show the results of different assessments using this tool. First, we show the results obtained during the course corresponding to the first ordinary call in the First Examination Date (FED). Next, we compare the call in June, with September and December, realized at September (Second Examination Date, or SED) and December (Third Examination Date, or TED), respectively.

### A. *First Examination Date (FED)*

At the beginning of the course all the material necessary to implement the course project is delivered, including the requirements. Along the course, four evaluations have been established for the assessment of the course project.

Once an assessment has been passed the student exceeds that matter. The student must realize one of those exams. The project must pass the quality criteria set by the tool TVSP. A student who passes the exam will not have to make another project delivery. A student who does not exceed the exam can present to the next call.

The first delivery was made in March, the second in April, the third in May and finally the last in June 2011.

Below are shown the results of the evaluations performed for the first call in the First Examination Date (FED).

We present the results of students who pass the assessment (or Student Passing PS) and students who failed the assessment (Not Passing or NPS). Table I shows the results of PS compared NPS.

TABLE I. RESULTS IN THE COURSE

| | **FED** | **SED** | **TED** |
|---|---|---|---|
| PS | 75 | 13 | 9 |
| NPS | 11 | 13 | 2 |
| Total | 86 | 26 | 11 |

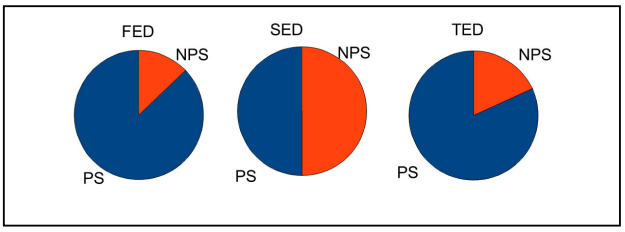


Figure 3. Results in the Course (percentage)

These results are shown for the four deliveries for the first call (First Examination Date EDF), and four scheduled dates in March (First), April (Second), May (Third) and June (Fourth).

The last row (Total) shows the total number of students who have presented to the call of March (First column), April (Second column), May (Third column) and June (Fourth column). According to the results shown, in all the assessments the total number of students who passed the examination was over 60% of students presented, even in the FED's third call the percentage of successful students is 100%. This shows that the student deliveries the course project when it has the minimum quality. These results are shown graphically in Fig. 4.

### B. *Results of the Course*

Table II shows the results of the three calls made along the course 2011 in June (EDF), September (SED) and December (TED) are shown. This table shows the results of PS compared with NPS.

According to the results shown in the three calls, a high percentage of students have overcome the assessment. This implies that, the student knows when the project has a minimum quality to successfully overcome the evaluation. Figure 5 shows that the percentage of success (PS) is higher in all evaluations to the percentage of fail (NPS).

This can also be seen graphically taking into account the total number of students presented to each of the calls (June, September and December 2011).

Figure 5 shows that the total number of students presented to the first call (EDF) is quite high compared to the rest of calls. This is because the course has facilitated and eased project delivery.

## VII. CONCLUSIONS

In this paper we have presented a testing tool used in FPII laboratory of Telecommunication Technologies Engineering Degree at University of Seville for the evaluation (and self-evaluation) of the student project. This tool allows autonomous verification by the student that can evaluate independently the proper execution of the course project. The results have shown that the use of this tool along with the flexibility to deliver the project to the teacher leads to a high degree of success.

TABLE II. RESULTS IN FED.

| | **FED** | | | |
|---|---|---|---|---|
| | ***First*** | ***Second*** | ***Third*** | ***Fourth*** |
| PS | 27 | 8 | 31 | 9 |
| NPS | 3 | 2 | 0 | 6 |
| Total | 30 | 10 | 31 | 15 |

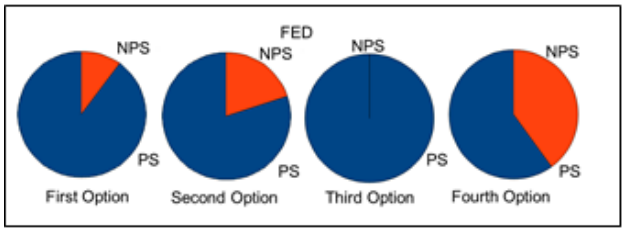


Figure 4. Results in FED: the four options.

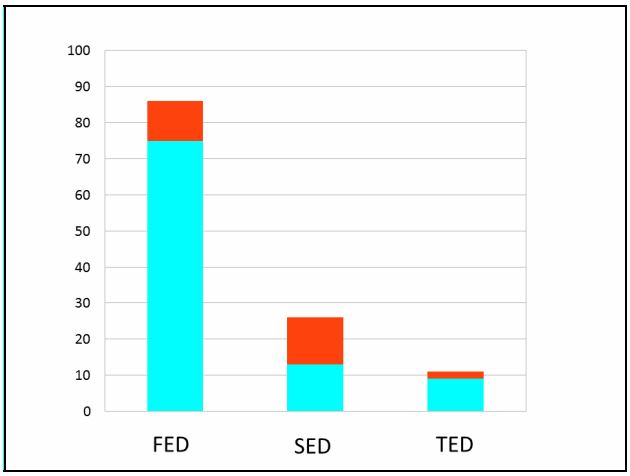


Figure 5. Results In The Course (total)

## ACKNOWLEDGMENT

The authors would like to thank to all teachers of the both courses (FPI and FPII) Rafael Bachiller Soler, Isabel Román Martínez, Antonio José Estepa Alonso, José Manuel Fornés Rumbao, José Ángel Gómez Argudo, Fernando Cárdenas Fernández, Pablo Nebrera Herrera, Godofredo Fernández Requena. These results would not have been possible without their dedication.

## REFERENCES

[1] K.M. Ala-Mutka, "A survey of automated assessment approaches for programming assignments". *Computer Science Education* 15(2):83-102, June 2005 http://dx.doi.org/10.1080/08993400500150747

[2] "Bologna beyond 2010 – Report on the development of the European Higher Education Area, Background Paper for the Bologna Follow-up Group", prepared by the Benelux Bologna Secretariat -, Leuven/Louvain-la-Neuve Ministerial Conference, 28-29 April 2009.

[3] BOSS Online Submission System. Department of Computer Science, University of Warwick, UK. http://www.dcs.warwick.ac.uk/boss/

[4] L. ChanLin, "Technology integration applied to project-based learning in science". *Innovations in Education and Teaching International*, 45, 2008, pp. 55-65. http://dx.doi.org/10.1080/14703290701757450

[5] Cheang, B., Kurnia, A., Lim, A., & Oon, W.-C. "On automated grading of Programming Assignments in an academic institution". *Computers & Education*, 41, 121 – 131, 2003. http://dx.doi.org/10.1016/S0360-1315(03)00030-7

[6] Curator: An Electronic Submission Management Environment. Department of Computer Science, Virginia Polytechnic Institute and State University, USA. http://courses.cs.vt.edu/curator/.

[7] Douce, C., Livingstone, D., and Orwell, J. "Automatic test-based assessment of programming: A review." *J. Educ. Resour. Comput.* 5, 3 (Sep. 2005). http://dx.doi.org/10.1145/1163405.1163409

[8] English, J. "Experience with a computer-assisted formal programming examination". *In Proceedings of 7th annual conference on Innovation and technology in computer science education*, 2002.

[9] Jackson, D.. "A semi-automated approach to online assessment". *Proceedings of 5th annual conference on Innovation and technology in computer science education*, Finland, 164 – 167, 2000.

[10] Jackson, D. & Usher, M. "Grading Student Programs using ASSYST". *In Proceedings of the 28th ACM SIGCSE Conference on Computer Science Education*, pp. 335–339. ACM Press, New Orleans, LA, 1997.

[11] Higgins, C., Hergazy, T., Symeonidis, P., & Tsinsifas, A. "The CourseMarker CBA system: Improvements over Ceilidh". *Educa-*

*tion and Information Technologies*, 8, 287 – 304, 2003. http://dx.doi.org/10.1023/A:1026364126982

[12] IMS Global Learning Consortium. (2005). IMS Question & Test interoperability specification (Version 2.0). Retrieved March 29, 2012, from http://www.imsproject.org/question

[13] Ley Orgánica de Reforma Universitaria. Ley Orgánica, 11/1983, de 25 de agosto. B.O.E. de 1 de septiembre.

[14] M. López, F. Gomez-Estern, and D. Muñoz, "Automatic Web-Based Evaluation of C-Programming Exercises in Engineering Education". *International Journal for Knowledge, Science and Technology*. Vol. 1. Núm. 2. 2010. Pag. 1-6.

[15] Morris, D.. "Automatic Grading of Student's Programming Assignments: An Interactive Process and Suite of Programs". *In Proceedings of the 33rd ASEE/IEEE Frontiers in Education Conference*, S3F-1 – S3F-5. 2003.

[16] IEEE Std 1003.1, 2004 Edition. http://pubs.opengroup.org/onlinepubs/009695399/mindex.htmll

[17] Perl, http://dev.perl.org/perl5/.

[18] J. Rué, *El aprendizaje Autónomo en Educación Superior*, Narcea, S.A. Ediciones, 2009.

[19] University of Sevilla, http://www.us.es.

[20] VirtualBox, https://www.virtualbox.org/

[21] VMware virtualization, http://www.vmware.com/.

[22] WebCT (Course Tools) or Blackboard Learning System, http://www.blackboard.com/.

[23] Y.T. Yu, C.K. Poon and M. Choy, "Experiences with PASS: Developing and Using a Programming Assignment aSsessment System", *Proceedings of the Sixth International Conference on Quality Software* (QSIC'06) 2006.

## AUTHORS

**A.J. Sierra** is with the Department of Telematic Engineering at the School of Engineering of the University of Sevilla, Camino de los Descubrimientos S/N, 41092 Seville, Spain (e-mail: antonio@trajano.us.es).

**T. Ariza** is with the Department of Telematic Engineering at the School of Engineering of the University of Sevilla, Camino de los Descubrimientos S/N, 41092 Seville, Spain (e-mail: matere@trajano.us.es).

**F.J. Fernández** is with the Department of Telematic Engineering at the School of Engineering of the University of Sevilla, Camino de los Descubrimientos S/N, 41092 Seville, Spain (e-mail: fjfj@trajano.us.es).

**G. Madinabeitia** is with the Department of Telematic Engineering at the School of Engineering of the University of Sevilla, Camino de los Descubrimientos S/N, 41092 Seville, Spain (e-mail: german@trajano.us.es).

This article is an extended version of a paper presented at the International Conference IEEE EDUCON2012, held in April 2012, in Marrakesh, Morocco. Received, 22 April 2012. Published as resubmitted by the authors 30 April 2012.