Importance of Open Source Contributions within the Educational Process

Victor Cărbune
Computer Science
ETH Zürich, Switzerland
vcarbune@student.ethz.ch

Laura Mihaela Vasilescu
Computer Science & Engineering
University POLITEHNICA of Bucharest, Romania
laura.vasilescu@cti.pub.ro

***Abstract*—Contributions to Open Source projects make an important difference in the educational process. By studying the underlying architecture of an Open Source application, students can significantly improve their technical knowledge and, also, contribute to the evolution of the project. Our research relies on an extra-curricular course that aims to speed up the process of the first upstream contribution. In this paper we highlight the impact of the course on the professional background and career path of admitted students.**

Keywords—Open Source; education; community; course

I. CONTEXT AND MOTIVATION

The university and academic environment already have numerous opportunities and responsibilities for students. However, several key elements that accelerate the process of contribution to real-life projects are missing or are simply presented too late in the university curricula. Open Source projects are the best way to broaden the knowledge within a particular Computer Science domain. They offer guidance to the internals of the project (code, architecture, design, etc.) and they also have a community behind that is keen to help enthusiast people.

Contributions to Open Source projects enhance the students' ability to easily adapt to a new project, to write beautiful and qualitative code and to easily understand and design an application in order to fulfill the needs of numerous users. The most important benefits of an Open Source contribution are the fast feedback one can receive from its community and the real-life applicability of the patch.

The Community and Development Laboratory [1] is a 10-week long course built by the Romanian Open Source Education (ROSEdu) [2] community. Through it, students gain extensive knowledge of the Open Source philosophy, of the various Open Source tools that any engineer should be aware of. By the end of the program they have made their first upstream contribution within one of the projects to which we managed to assign mentors.

We concentrated the course activities along the following four components within the Open Source framework: the philosophy, the community of a project, the development model and the resulting software itself.

The created environment accelerates the regular process of submitting a contribution, by bringing together non-experienced students and active contributors within a wide variety of Open Source projects as mentors.

This paper studies the impact of Open Source contributions on the students' career path, considering the formal environment created through the course. It is interesting to understand in what manner the particular changes brought to the program have been perceived by different generations of students.

II. HISTORY: THE FIRST EDITION

The first edition of the program took place in the spring of 2009 and it was received with enthusiasm and interest by both graduate and undergraduate students. From the total pool of 102 applicants, 16 participants were selected to work on four projects. The main components of the course were technical presentations and mentorship sessions. The projects were written from scratch, unlike the traditional path of an Open Source contribution, by teams of four students and one or two mentors.

Students gained multiple skills through the course. They learned how to work in a team, how to plan a project and how to design the architecture of an application. At the end of the course, working with a version control system and understanding code written by somebody else ceased to be a mystery.

The first edition focused on sharing knowledge about the tools one can use in software development, especially in Open Source projects. The following topics were covered: *History and Philosophy of Open Source*, *Motivation of Open Source*, *Editors (vim)*, *Version Control Systems* (Subversion and git), *GPG*, *SSH and Public Keys*, *Bug tracking*, *Project Hosting*, *Architecture and Design*, *Project Planning and Licensing*. According to the feedback received from the participants, it was definitely a successful edition. However, several key aspects had to be changed in order to facilitate students with the proper environment of submitting upstream Open Source contributions during the course.

To start with, we realized that building a project from scratch doesn't give one the opportunity of interact with a

community. One of the most important aspects in making a contribution to an Open Source project is the actual interaction one can have with other developers and this was a missing component. In addition, the feeling of having your code submitted upstream and being used by others can never be achieved by working at a completely new project, built from scratch. During a semester course, the project hardly evolves quickly enough in order to become the new Linux kernel or it simply goes wrong in many unpredicted ways, due to lack of experience.

Another important aspect is that in different Open Source projects, communities use various tools for distributing or reviewing the code. Some might use git, others might use Subversion and so on. An important question appeared: how can we design the course in order to have a general character, but, in the same time, to be specific enough in order to satisfy as many Open Source project requirements as possible? To accomplish this, we realized that it is important to teach the basic skills required in both a theoretical and a practical manner. If there are several tools fulfilling the same function, then the best option is to teach one of them and to teach it well.

Following the principle *less is more*, we decided to change the course experience significantly (as can be seen in the next chapter).

III. CHOPS AND CHANGES

The second and third editions of the course have brought major changes, aiming to shape the experience of the participants to be as close as possible to the normal path of submitting an Open Source contribution. This involved seeking mentors, that are actively involved in public Open Source projects, to coordinate students and, together, enhance pre-existing software, such as KDE SC or Gnome, with a new feature, rather than create a completely new project.

There are extensive benefits and challenges of such an approach, closer to the real Open Source world, such as the needs to:

- familiarize with a new codebase
- understand the general guidelines and plans for building new features
- integrate within an existing world-wide community

Starting from the second edition, the overall structure of the course was changed to emphasize the development of soft skills and to create focused development sessions. From the feedback received, it seemed that some courses didn't have an important impact for their learning curve and therefore we adapted the curriculum of the courses to better prepare students for their first contributions, by adding or dropping some of them.

Transmitting technical knowledge to students is not enough for their personal development and for their future careers. We believe that students can learn a lot from successful professional experiences recounted by recognized people and therefore we have invited leaders from different companies and communities to give talks at several courses. Personal examples can be very motivating and encouraging in the educational process.

We thought more carefully on how students were spending their 4 hours during a session: we added a two-hour weekly hands-on activity that brought students together with their mentors to work on the projects. At the end of the course, this changes lead to a positive outcome. Students improve far more in the sessions spent together with their mentors, than by participating in a presentation of another tool that they might not even use during their project.

IV. ANALYSIS OF THE COURSE IMPACT

This paper is the result of the analysis of periodic feedback received in the last five editions and the personal evolution of the participants. In order to demonstrate the achieved results, we also designed two additional surveys:

- one completed by 65 people that didn't attend the course
- one completed by 53 people that attended the course

The surveys share some of the questions that are not directly related to the course. Their purpose is to determine, in general terms, the impact of Open Source contributions on the career path of recently graduated students.

A. *Participant Profile*

In order to give more sense to statistical data, we have gathered general information about the participants, their activities and prior interactions with the Open Source world before enrolling to the course.

From the second edition of the course, the target students are in their first or second year at the university and the average age of the participants is between 19 and 20 years old. Most of them are enrolled at University POLITEHNICA of Bucharest, Faculty of Automatic Controls and Computer Science.

According to the surveys, the interest for the course was triggered by the reasons presented in *Table 1*.

Reason	Percentage
To work in a team and to meet new people	81%
Interest in Open Source Philosophy	70%
Interest in the community behind the course	66%
Desire of pursuing extra-curricular activities	55%
Adding extra-curricular information in the CV	32%
Course reputation	30%
Desire of participating to an interview	30%
Interest in a specific project from the course	9%

TABLE I. REASONS FOR SUBMITTING AN APPLICATION TO THE COURSE

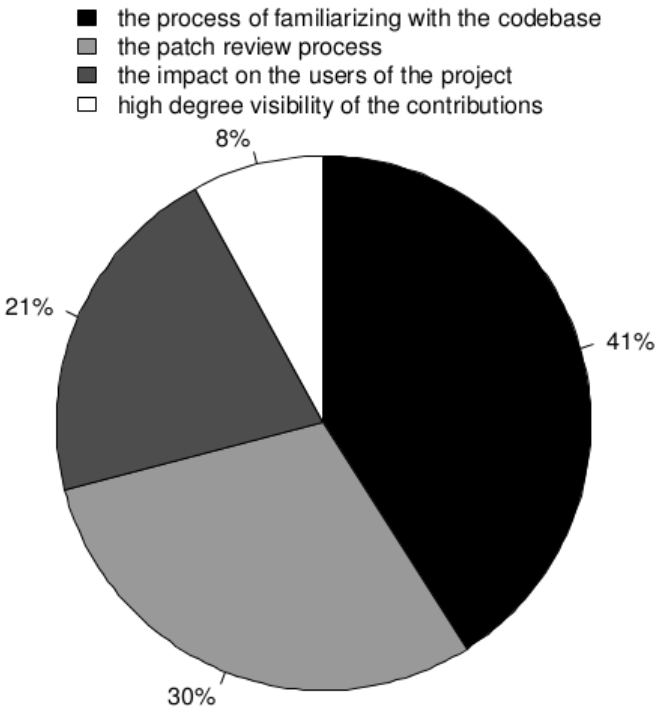

Fig. 1. Benefits of Open Source Contributions According to the Participants

Among the participants of the course, more than 94% of them have never made an Open Source contribution before attending the course. Therefore, most of them have done their first contributions during the course workshops.

B. Mingling with the Formal Education

The course curriculum and activities are closely developed in correlation with the academic calendar and deadlines of homework happening in the same time span. The reason behind this is that the skills developed during the course are complementary with the ones achieved in the academic environment.

It is important to maximize the gain from both environments and to find a balance between them. Respondents classified the time management during the course as follows:

- 45% completed both academic and course requirements successfully
- 40% had minor difficulties in completing both activities
- 13% were not able to finish to successfully finish some of assignments
- 2% had to completely withdraw from the course

An important result we found is that more than 90% of the participants think that the course curriculum has successfully accompanied the elements taught in the regular academic lectures and laboratories.

Teamwork is a highly encouraged component of the course and an essential element in the undergraduate level education. One of the elements that we believe to be extremely hard to guarantee is the fairness and equal distributions of tasks among the different members of the team. Another challenge is to ensure similar technical background among the students. We have achieved this through a competitive admission process, rated by more than 58% of participants as highly competitive.

The collaboration experience within project teams is considered a successful component. More than 73% of the participants rated their team mates' technical expertise similar to theirs and more than 83% appreciated that the tasks were uniformly distributed among each member of the team they were part of.

According to the participants enrolled in the course, the most important benefit of Open Source contributions can be seen in *Fig. 1*.

C. Longterm Impact of Open Source Contributions

We have questioned the respondents about the impact of Open Source contributions done during the course on their overall academic performance. More than 90% of the respondents thought there was a positive effect on their knowledge skills.

There is a common subset of skills that are easily gained through the process of making Open Source contributions. We have used the two different surveys in order to understand to what degree the course has successfully developed these skills.

In *Fig. 2*, the two types of bars represent relevancy percentages for the two categories of respondents. The demanded skills represent the percentage of questioned persons that consider the respective skill as something that should be extensively developed throughout the university.

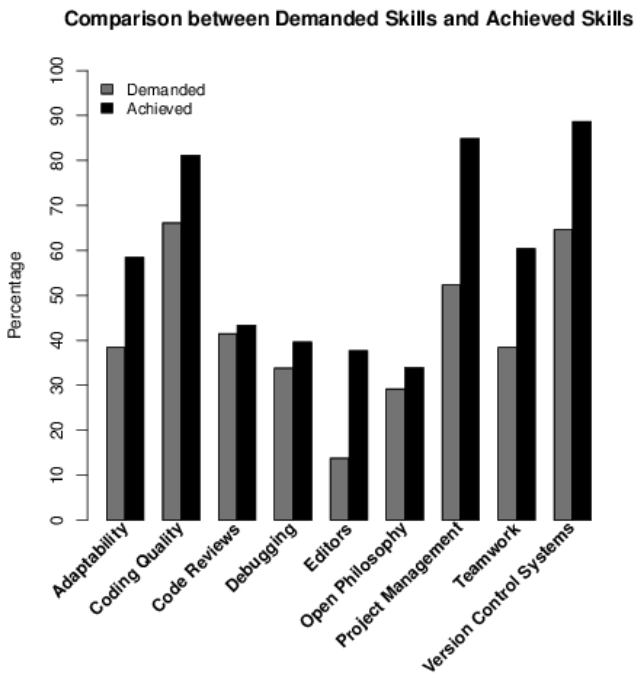

Fig. 2. Comparation between Demanded Skills and Achieved Skills

The achieved skills represent the percentage of students that attended the course and consider that the respective skill was greatly enhanced throughout various components of activity.

In the final section of the survey, we asked the participants whether the course accelerated the process of becoming a contributor within an Open Source project and whether the Open Source contributions done by the participant had any influence on receiving a job or internship offer from a company in the software engineering industry.

We were interested in understanding how many of the course participants became contributors to an Open Source project and when exactly. The survey results can be analyzed in *Fig. 3*.

When asked whether the participation to this course had any impact on the process of becoming a contributor to an Open Source project, 70% of the participants responded affirmatively, which coincides with the number of actual contributors. The participants agreed unanimously that such a course has a direct impact on becoming a contributor.

Obtaining a job or an internship position is one of the success criteria for an educational program. The employment rate among the participants of the course is over 90% and therefore we questioned the survey respondents whether the course or the Open Source contributions they made (even after the course ended) had an impact on obtaining the offer.

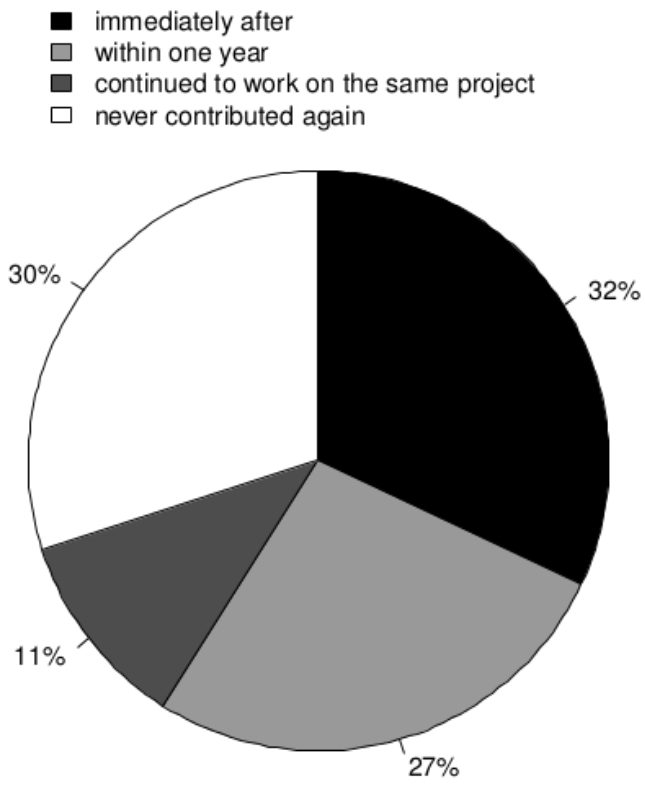

Fig. 3. Timeline of devoting to Open Source

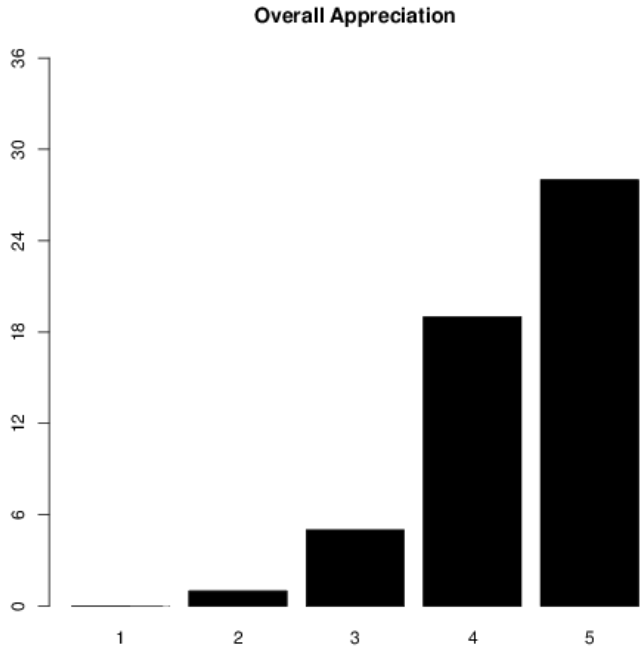

Fig. 4. Overall Appreciation

According to more than 71% of participants that obtained a job or internship offer, the course has had a positive impact. The stated reasons are related either to the visibility of the Open Source contributions or to the technical and non-technical skills that they have gained during the course.

Asked whether they would enroll again to such a course or not, 96% have responded with *yes*. The overall utility of the course is depicted in *Fig. 4*.

V. RELATED WORK

In an empirical study [3], multiple hypotheses about Open Source software have been analyzed through an empirical study. We consider most of the conclusions important from an educational perspective.

Creativity is a key element in the formation of an individual and, according to the study, evidence has been found to support the fact that Open Source software fosters creativity. In the same time, no evidence has been found to support the success because of simplicity or a higher level of modularity than closed software. These results are important for the educational process, since a new graduate should be equally prepared to work with both types of software ideologies. Simply put, this means that an extended experience with Open Source software allows the contributor to deal with issues that are also commonly encountered in proprietary software.

Teamwork has proven to have a positive effect in terms of achieving the stated course objectives, when compared to the same performance of students working individually. The result is extensively treated in [4].

In [5] many of the challenges of incorporating the regular Open Source software developing model in the educational process are extensively presented, extending beyond the

elements we considered in this paper. The course we created is the first step towards a successful integration of the two. However, scaling it properly to a university wide course remains an open issue that we will further investigate.

VI. CONCLUSION

The impact of the course has been studied among five different generations. After each course a post-performance analysis was made. The analysis and the feedback received from the students were then used to improve the course structure and it is presented in detail in the final paper.

The immediate impact of the course was on the students' decisions and opportunities: many of them have continued their Open Source contributions, participated at conferences and development camps, others obtained internship and job offers within important companies or even started their own companies. More importantly, their work often followed the Open Source philosophy.

ACKNOWLEDGMENT

We would like to thank to Alex Eftimie and Lucian Grijincu (the first edition coordinators), Mihnea Dobrescu-Balaur and Andrei Petre (the fourth and fifth edition coordinators), Răzvan Deaconescu (that came with the original idea) and, last but not least, to the community of the Romanian Open Source Education Association[2] (ROSEdu), who made everything possible.

REFERENCES

[1] http://cdl.rosedu.org/english

[2] http://rosedu.org

[3] Paulson, James W., Giancarlo Succi, and Armin Eberlein. "An empirical study of open-source and closed-source software products." *Software Engineering, IEEE Transactions* on 30.4 (2004): 246-256.

[4] Oakley, Barbara A., et al. "Best practices involving teamwork in the classroom: Results from a survey of 6435 engineering student respondents." *Education, IEEE Transactions* on 50.3 (2007): 266-272.

[5] Meneely, Andrew, Laurie Williams, and Edward F. Gehringer. "ROSE: a repository of education-friendly open-source projects." *ACM SIGCSE Bulletin*. Vol. 40. No. 3. ACM, 2008.