Teaching Software Development in Community-Driven Software Projects: A Practical Experience*

Gregorio Robles
GSyC/LibreSoft, Universidad Rey Juan Carlos
grex@gsyc.escet.urjc.es

Santi Caballe
Universitat Oberta de Catalunya
scaballe@uoc.edu

Jesús M. González-Barahona
GSyC/LibreSoft, Universidad Rey Juan Carlos
jgb@gsyc.escet.urjc.es

Abstract

This paper presents a teaching experience at an on-line master at the Universitat Oberta de Catalunya. For the final course, which leads to the presentation of a master thesis, students have to set up a libre software project with the aim of making its development community-driven. This means that besides common programming activities, efforts have to be devoted to obtain feedback from users, boost the participation of other programmers, impact the whole libre software community, among others. The course has been scheduled so far in the last four semesters with over a dozen students taking part with a high rate of success.

1. Introduction

Libre software has gained in recent times a major attention in industry and academia. Due to its importance and impact on industry over the last years, higher education institutions have adopted the development of libre software in their academic programs. However, in developing real libre software projects, several key issues are hard to be reproduced in traditional academic curricula, such as a community-driven approach. This is an essential aspect to both engage and

*The work of Gregorio Robles and Jesús M. González-Barahona has been funded in part by the European Commission QUALOSS (FP6-IST-5-033547) and QUALIPSO (FP6-IST-034763) projects.

manage contributors in the project. Indeed, libre software projects typically consider a sense of ”community” in the software engineering process, that involves important and complex issues, such as intellectual property and management of external contributions, as the very rationale of this phenomenon.

Despite many post-graduate degrees have been proposed and implemented so far on libre software in quite a few high educational institutions all over the world, their curricula have shown important barriers and deficiencies. To the best of our knowledge, there is no academic experience that provides students with theoretical and practical expertise on how to manage a project, to design and set-up the required infrastructure, to attract and manage external contributions, etc. The closest efforts to this direction are certain university courses where students are invited to join libre software projects and to collaborate with them. Even though this represents an important step toward this direction, we think this is not sufficient.

In general, academic programs addressing libre software can be classified into two groups: those that focus on the philosophical, social, economic and management issues and those that are more concerned with the technical parts of the phenomenon, providing technological insight into libre software technologies. The first group is basically theoretical whereas the second presents the most important libre software products, such as Apache, GNOME, and KDE. Despite the technical skills that are required to work with these technologies are involved, these programs usually lack of any interaction with the community.

This paper presents a novel approach for teaching libre software development that the authors have introduced at the Universitat Oberta de Catalunya (UOC). The UOC (http://www.uoc.edu) is located in Barcelona, Spain, and offers full distance education through the Internet since 1995. The virtual campus supports currently about 40,000 students, lecturers and tutors who are involved in 600 on-line courses from 23 official degrees and other PhD and post-graduate programs. The experience reported here is based on an on-line master course at the UOC as part of the curricula of an official post-graduate degree on libre software, that has been put in practice in five semesters with around 30 students over the last two years. As part of this degree’s curricula, students have to perform a master thesis course that accounts for 15 ECTS. ECTS stands for European Credit Transfer and Accumulation System, which is for measuring and comparing the study burden and attainment in higher education across the European Union. One ECTS accounts for 20 to 30 hours of learning, including reading materials, class discussions, lectures, homework, exams, etc.). Therefore, the proposed course accounts for 300-450 hours of effort on the student side.

The paper is structured as follows: Next the goals of the subject will be presented, basically the competences that a student should acquire in it. Then, we will proved some insight into the libre software master programme at the UOC and the specific outline of the master thesis subject. Finally, our experience is presented and conclusions are briefly drawn.

2. Goals

The main goal of the master thesis subject is that students experience how to manage a libre software project in a *real-world* environment, ranging from technical activities such as development to social and marketing efforts to set up a community around the project. This includes philosophical, technical and marketing aspects as well, so the selection and discussion of the chosen license, the

set up of a proper infrastructure to the planning of a strategy to attract the interest of users and other developers are part of the work that the student has to undertake. Once the course is over, the student should be completely familiarized with the tools used for collaborative development in the libre software world, those used for the communication exchange of the various agents that participate in it, and have a basic knowledge on how to organize and publicize new releases of the project. From this knowledge, students should be able to create and launch their own software project or to become part of an existing one.

The subject uses a textbook that has been written exclusively for this course [1]. Currently it is only available in its Spanish version, although a Catalan and an English version are in consideration in the next future.

The method used in the course is based on some principles that have been reported in many sources, most notably [3, 2]. Users have to be attracted and maintained in the project; thus, users should have a low barrier of entry to install and use the software and participate in the user community. A minimum level of activity and diffusion about the novelties in the project should be given. The community of users and developers should be fostered to have an autocatalytic process where we do not need to fulfill most of the tasks. The automation of tasks is a key point in the management of the project as it will lower the administration tasks.

3. Academic context and outline

This subject is part of the official master of libre software at the Universitat Oberta de Catalunya. It is a 60 ECTS credits master with following curriculum:

- 20 ECTS credits correspond to mandatory subjects: introduction to libre software, basic use of GNU/Linux, advanced administration of GNU/Linux and implantation of libre software systems,
- 25 ECTS credits for specialization subjects. Students have to choose one of the following four specialization tracks:
  - administration of networks and operating systems in libre software environments,
  - administration of web and electronic commerce in libre software environments,
  - management of information systems in libre software environments, and
  - development of libre software applications.

  The track that our approach belongs to is the one devoted to the development of libre software applications. Among the subjects that are recommended in this track for it we can find “introduction to software development”, “advanced concepts of software development” and “software engineering in libre software environments”, each of them having 5 ECTS. These subjects can be complemented with others from the other tracks, such as databases, legal aspects, etc. in order to achieve the required 25 ECTS.
- and 15 ECTS credits for the master thesis course.

To complete the course successfully, the student has to have in advance some previous knowledge referred to the philosophy of the libre software movement, basic knowledge of the development process, some insight about the various licensing schemes that exist and the most important software projects that exist. Most of these contents, if not all, have been acquired in previous subjects of the master.

Regarding technical aspects, students have to be familiarized with a GNU/Linux environment and have some specific knowledge about the development tools. It is important to note that this course will not be a programming course. It is supposed that the student has some programming notions to create software. The aspects that will be of most importance will handle with how to achieve a software project that enters into a process of attracting new users and developers and that creates a community around it that will develop the software in a collaborative way.

The subject can divided in two main parts: first a theoretical part followed by a practical one. First, the collaborative development tools commonly used in the development of libre software projects are presented. As there are some web sites that offer them without cost for projects with free license, the use and inner working of these sites is presented. Then, the use of a source code management system is introduced. This will be done according to two views: as administrators of a repository we will see how to install, configure and manage user accounts. From the point of view of a user, some advanced elements will be introduced in addition to the basic functioning.

Then, bug management tools will be presented, in particular the most used one in large libre software projects: Bugzilla. As in the case of the source code management systems, the set-up of such a system will be shown, in case the student wants to be the administrator as well. Finally, mailing list managers are presented, in particular GNU Mailman.

The last part of the outline is the most important. It is a guide with the steps that have to be followed to achieve a community around a software project. Lessons learned and good practices proposed have been obtained from real libre software projects. This part can be divided in two, as well: one devoted to the set-up and launch of the project and a second one related to the continuous effort required for its management.

4. Experience and conclusions

This subject has been scheduled in the last four semesters with over a dozen students taking part in it with a high rate of success. Most of the problems that students raised have been due to the fact that the goals do not depend directly on their own activity, but on decisions taken by third persons. This is because the impact on the community has been shown to be a very difficult task to achieve in a short period of time. Students feel that the control they have on the output is low, producing stress and uncertainty and might affect their productivity. To avoid this, we recommend lecturers to encourage students during the whole process and to let them know that they will grade not only the output, but also the approach that the student has taken.

One of the major problems that we have found with our approach is that the amount of effort required by a student is high. Hence, the course is now offered in two semesters, with 7.5 ECTS each. This has also the positive effect that there is more time to plan activities for a longer period of time and to obtain more feedback from the community.

In summary, we have seen in this short paper an experience of teaching community-driven libre software projects. The main point of our approach is that the development activity is not the main part of the effort to be performed by the student, but that other community-boosting actions have to be considered, planned and undertaken.

References

[1] J. J. Amor, I. Herraiz, and G. Robles. *Desarrollo de Proyectos de Software Libre*. Fundación Universitat Oberta de Catalunya, May 2005.

[2] B. Behlendorf. Open source as a bussiness strategy. In C. DiBona, S. Ockman, and M. Stone, editors, *Open Sources. Voices from the Open Source Revolution*. O'Reilly & Associates, 1999. `http://www.oreilly.com/catalog/opensources`.

[3] E. S. Raymond. The cathedral and the bazar. *First Monday*, 3(3), March 1998. http://www.firstmonday.dk/issues/issue3_3/raymond/.