PicoLibre : a free collaborative platform to improve students'skills in software engineering

Eric Cousin, Gérald Ouvradou, Pascal Pucci, Samuel Tardieu

PicoLibre: a free collaborative platform to improve students' skills in software engineering

Éric Cousin, Gérald Ouvradou, Pascal Pucci, Samuel Tardieu

Groupe des Écoles des Télécommunications
46, rue Barrault
75634 Paris Cedex 13, France
Eric.Cousin@enst-bretagne.fr, Gerald.Ouvradou@enst-bretagne.fr,
Pascal.Pucci@enst-bretagne.fr, Samuel.Tardieu@enst.fr

Abstract— **The work described in this paper consists in the development of a pedagogical collaborative platform to improve the know-how of our students in software engineering and collaborative work. Its use in a free software context leads to greater commitment of the students and better reusability of their work.**

Keywords— **Skill development, new information technology tools, project activity, active learning, monitoring, sofware engineering, collaborative work, free software**

I. Introduction

WHILE widely used in graduate or post-graduate French scientific cursus, project activities often miss some fundamental points. In particular, technical goals tend to prevail too heavily on organizational concerns. That is how, for instance, Computer Science students are trained to build software from scratch by themselves rather than to reuse existing code and to produce reusable code collaboratively. Yet, reusability has proven to be the most rational approach: it saves time and leads to more reliable software. Free Software practice relies heavily on this approach and that is where it takes place in our project.

In our view, opening projects to the Free Software community and using adequate collaborative tools can correct the forementioned bias. Work described in this paper consists in the development of a pedagogical collaborative platform, called PicoLibre. Its integration within our cursus helps to improve students' know-how in collaborative software engineering.

In the next section, the general context and the pedagogical needs are given. Third section deals with PicoLibre: a brief history of the project is made, and the main features of the platform are described. The fourth section explains how the use of the platform is promoted in our cursus, and makes some statements about the pedagogical impact. Crucial role of free software is underlined and some perspectives are drawn in the final conclusion.

II. Context and pedagogical goals

The Groupe des Écoles des Télécommunications is composed of three main engineering schools and research centers. Highly-skilled students are selected by competitive examination and will be awarded a master's degree after three years of intensive study. We also have some hundred new PhD students every year.

During their cursus, all students have to complete several team projects in various fields (computer science, electronics, physics, ...). Each teacher will favour projects close to his research interests while students wish to achieve something they will be proud of.

According to French law, any software created by the students belongs to them, even if they have used college computing resources. To be able to use what was developed under our supervision for our research needs, we can request students to make a version of their work available under a Free Software license. Such a license grants the recipients (supervisors) the following freedoms[1]: *"the freedom to run the program, for any purpose; the freedom to study how the program works and adapt it to one's needs; the freedom to redistribute copies; the freedom to improve the program and release one's improvements to the public, to the benefit of the whole community."*

While using such a Free Software license preserves students' rights over their software (including the right to make a proprietary closed-source version if they want to), it gives supervisors the permission to use, enhance and distribute project results. Of course, unless the license allows it specifically, if a redistribution of the modified versions is made, it has to use the same license as the original, granting the recipients the same rights as given in the first place. The GNU General Public License (GPL) is one of those Free Software licenses. Any work derived from a software protected by the GPL cannot be released under a different license.

Our students are taught software engineering techniques. Typically, they have to accomplish certain pre-

defined milestones in their project and defend their choices (planning, problem analysis, solution architecture, testing and conformance checking). In the best cases, they are required to use configuration management tools such as CVS[1] in order to maximize team work efficiency and change tracking through the use of carefully thought-out commit messages.

However, this leads each group of students to use its own private repository[2]; at the end of the project, the supervisor only receives the final version, and all the history information which has been conscientiously collected and used throughout the project development is then lost. Most of the work done in a project is therefore never reused nor even integrated in a larger framework. Projects tend to terminate near the end of the semester, when the teacher is busy correcting exam papers and evaluating project results. Moreover, the missing history information makes it virtually impossible to have another team of students start another project on top of the existing one. For the same reasons, starting development of a larger Free Software project based on these grounds is difficult.

In fact, what we need is a system which supports both active project development and project memory facilities. We want not only to provide project development hosting, but also mailing-list features so that a new member can reconstitute the whole project evolution over time. Our view is that documentation and WWW pages associated with the project deserve the same treatment as the code itself, both during active development and between active phases. All these features constitute the aim of the PicoLibre platform which will now be described.

III. The PicoLibre project

The first step of the project involved the technical specification of the platform. To this end, we studied existing hosting platforms[3] looking how they could match our specific needs. Beyond this, our goal was also to look at the different technologies commonly used in such platforms to underpin our reflection.

We stated that our platform ought to have four major features. It should be:

- simple to use;
- simple to install;
- simple to manage;
- available in its entirety under the GPL.

It should be "simple to use" because, for a new user, the investment involved in learning platform usage has to be as light as possible compared to the code development load. Even for projects of moderate size, we want to promote the use of the platform, thus the induced overhead must remain at a reasonable level.

It should also be "simple to install" and "simple to manage": we wish to exchange platform experiments with other engineering schools and universities with the aim of improving our tool as well as our pedagogical methodology. Thus, the platform has to be suitable for adoption by light structures without heavy computer support.

The GPL license indeed makes it easier for anybody to use the platform, while it guarantees its durability: the software will never be closed, thus anybody may benefit from successive improvements.

Keeping this in mind, we mainly examined "Sourceforge"[2], the most famous Free Software hosting platform. Today, Sourceforge hosts as many as 30,000 open source[4] projects and 10 times more users. This platform is based on PHP technology[5]. Sourceforge seemed to us obviously an excellent groupware system for confirmed developers but, for our purpose, the training investment appeared far too great. In fact, Sourceforge offered too rich a framework for a novice and, moreover, it appeared also that it did not meet our two other criteria. Besides an extremely complex installation process, we noticed poor modularity of the software. This comes probably from the fact that the system had grown over time and was not, at the beginning, structured to become such a huge system. Trying to go back towards a simpler version of the platform does not provide a simple way to recover a simpler architecture.

Thus, the idea to try to adapt Sourceforge to our needs was forgotten. We therefore decided to look for another groupware kernel as the basis of our platform, which better matches our criteria.

From an architectural point of view, such a system is made up of two different parts: the *frontend* and the *backend*. The former has to support all user interactions typically through a web-interface. The latter is in charge of the project and user management. Basically, a platform hosts a set of projects. With each of them is associated a group of users. Each user has specific access rights to the various components of the project which he can manipulate thanks to dedicated tools. Technically, the backend implementation mainly relies on a database management system.

After analysing several groupware systems, we decided to adopt the PHP-GroupWare framework[3] as the basis of our platform frontend. PHP-GroupWare (PH-

[1] CVS (Concurrent Versions System) is the most commonly used version management tool in the Free Software world.

[2] The repository is the place where files are stored permanently, along with their full history.

[3] A hosting platform offers a set of computer resources in order to store all the components of a project (including user accounts), and tools for facilitating cooperative work.

[4] "Open Source" is a less restrictive definition than "Free Software", see `http://www.gnu.org/philosophy/free-sw.html`

[5] PHP is a widely-used general-purpose scripting language that is especially suited for Web development and can be embedded into HTML, see `http://www.php.net/`

PGW) is a free multi-user web-based groupware suite written in PHP. It also provides an API for developing additional applications. The greatest benefit we got with PHPGW was the high level of abstraction it offers for programming, thus speeding up and making our work more reliable. However, as PHPGW was based on a "per user" paradigm instead of the required "per project/per user", we had a significant amount of work to integrate this new paradigm into PHPGW. This will lead to a patch to be produced as our contribution feedback to this Free Software project.

In a cooperative and concurrent software development process, one of the most critical functions is the code versioning management. In our platform, this backend function is supported by CVS which is known to be a very secure system. But, this is insufficient if user access to the code repository is not strictly controlled. To this end, we adopted a Free Software produced by IDEALX which offers secure (i.e. crypted) web access to the CVS repository[4]. As we used LDAP[5] to register users on our platform, we had to adapt the IDEALX product. This add-on has been recently submitted as a patch to the project development group.

The first production version of our hosting platform PicoLibre was made available at the end of 2001. Here are the main features offered to a user registered in a hosted project group:

- Code and WWW repositories are under control of CVS and are accessed through web secured connection. Thanks to CVS, the history of these two data spaces is continuously traced, allowing them to be restored to any previous state. The WWW repository is used to feed the project web site. This site is activated as soon as the project is accommodated on the platform.
- Two mailing lists are automatically set up when a new project is hosted. Typically, one is used solely for developer communication and the other for both user and developer communication, but the usage remains up to the project group. Each group member may himself subscribe to or unsubscribe from the lists. These are continuously archived, thus the history of mailing group activity is permanently available for all members of the group. The mailing-list system manager is the Free Software Sympa[6].
- A bug tracking tool allows the complete cycle life of a bug - once detected - to be traced. A bug cycle life begins with a bug submission which can be made by anyone. The submission is then analysed by a member of the project developer team. If it is just a misunderstanding and not a real bug, the submission is rejected and may lead to a new item in a FAQ (Frequently Asked Questions). Conversely, a "real" bug is stated as open. When a developer deals with it, the bug becomes assigned. Then, the developer submits a patch intended to fix the bug. When the patch has been successfully tested, the bug is marked closed and is archived. The patch is then integrated into the so called "project development version"(as opposed to the "stable version").
- A task scheduler board allows a set of tasks concerning the project development to be scheduled and human and material resources to be allocated to them. For instance, a bug fixing task may be planned with this tool.
- Project documentation and downloading tools are also provided. Various web pages are automatically set up at the project's creation. To do so, questionnaries are submitted to the project creator. As soon as a first version of the code is available, it becomes accessible to authorized people (see below) through a downloading tool.

The administrator of the platform (user and project management) is offered several tools. For instance, visibility of a project is controlled via a private or public status. A *private project* involves a restricted access limited to a set of identified users. Conversely, all components of a *public project* are accessible to any registered user. There is also an anonymous access mode which allows to access to certain parts of a public project (e.g. the code downloading) without requiring to be registered on the platform. In a project user group, two categories of user are implemented: administrators and developers. Administrators may set up and modify at any time the access rights of other group's users.

Other functionnalities are currently under development. For instance: a documentation formatter based on XML, a search engine to browse projects hosted on the platform, a project activity ranking tool, a platform monitoring system, and so on. Of course, an important point to keep in mind is the pedagogical objective of PicoLibre. Each new tool considered has to be examined from this point of view (i.e. real need, real usefulness, impact on the user-interface, time needed to master this new functionality, etc.). To this end, PicoLibre architecture is modular and flexible, as it also allows easy customisation of each site installed. Soon, a Debian package of PicoLibre will be available and so, the installation process onto corresponding platforms will be dramatically improved. In the future, we intend to produce software packages for the most common Unix-GNU/linux based platforms.

IV. Integrating PicoLibre in our cursus : first results

We have been using our platform for several months now[7]. Users of the first prototype - whose name was Serveur Libre - were mainly the handful of students that were developing it. Nevertheless, this brought us enough feedback to confirm the previously identified needs and to design the new version of the platform, PicoLibre. This was soon judged stable and mature enough to enter operational use. Introducing PicoLibre into our cursus was therefore possible.

To remain in the Free Software spirit, we wanted to keep the same approach as the one followed since the beginning of the project. That is, students who take part in the development of the platform itself, or simply use it to host their project, should do so on a voluntary basis.

The first pedagogical challenge is therefore to encourage them to use the platform. Given that some software engineering courses are already taught in our cursus, only a short additional introduction to our platform is therefore required ; the main functionalities are presented in one class, and then practiced directly on the platform with some demonstration projects. As the overall use of the platform is simple, the keypoint lies in understanding the use of CVS; this is rather simple. Students then know enough about the platform to make their choice accordingly.

Taking into account observations based on the previous prototype and first feedbacks with PicoLibre, here are some findings and what we have learned about the pedagogical impact of the use of this platform:

- Good practice advice in software engineering is too often considered as obvious and theoretical by students. Programming on a small scale, as it is mostly the case in their cursus, generally does not allow them to realize how difficult collaborative development may be, and that some discipline should be followed. The use of the platform makes them have another outlook on these aspects and improves their practices.
- When hosted on the platform, students'work has a better chance to be really used. As an example, our project itself involved more than fifteen student projects that the platform allowed us to proprerly supervise and integrate.
- As we already stated in a previous article[8], opening their projects to the Free Software community has a great impact on the way students work. Knowing that everybody can access their work[6] increases their overall motivation and their commitment to produce reusable code.
- As far as evaluation of students' work is concerned, we take for granted that, most of the time, the students' code is not inspected by supervisors. Evaluation is therefore mainly based on the overall aspect of the software, and accompanying reports. From this point of view, the platform allows supervisors to follow more closely what is going on, but does not help very much. Furthermore, the underlying philosophy of PicoLibre is rather to encourage students to reuse existing code and to work together with other students. This raises a new difficulty. Part of this concern may probably be tackled with some new monitoring facilities to be added soon to the platform.

[6]Of course, only some of the projects meet all the requirements to really be valuable for the community.

Given the mentioned pedagogical interests, a midterm objective is therefore that each student should use the platform for at least one of his projects so as to fully understand the usefulness of the different tools offered, and to acquire know-how in collaborative engineering.

V. Conclusion and perspectives

We were surprisingly pleased by the success of our experiments. While we were quite anxious not to hamper students' project development because of bugs or inadequacies in our platform, we did obtain positive feedback and suggestions. The use of PicoLibre has shown good pedagogical advantages.

It is to be noted that Free Software plays a central role in our approach. We have just mentioned how sharing their code with the Free Software community may positively influence students' work. Another point is that development of the PicoLibre platform itself would not have been possible without the availability of all those software components described in section 3. Finally, PicoLibre is also a Free Software[7]. As it is self-hosting[7], anyone with internet access can get the latest version and install it on his computer to host public or private software development, WWW sites, internal documents, and so on. Anyone can also contribute by adding features, fixing bugs or writing documentation. This allows any other university to try the platform, and should favour the growth of the project.

The first publicly available release has been tested on Debian GNU/Linux systems. We are now extending the range of supported platforms; the first one on the list will be FreeBSD, another high-performance Free Software Unix system. Our goal is to run PicoLibre on any operating system where PHP Groupware (the web software framework we use), CVS, SSH and Sympa are supported; that includes virtually any Unix-like system.

Monitoring facilities are being added to the PicoLibre platform. These facilities will help a supervisor to analyse what is going on in a project and within a team. For example, the number of changes made by each team member will be available, as will be the number of features added. Also, the vitality of the project will be evaluated; this should ease early detection of blocking problems.

We have also been working with numerous partners on the CoopX[9] project to define a communication protocol allowing integrated development sites such as PicoLibre to exchange information[10]. For example, a developer registered on a PicoLibre platform could use any GNU Savannah site without registering again. Another concern is project migration: many people want to be able to move from one platform to another if they are not satisfied with the service quality or if they need

[7]Accompanying tutorials are also freely available

different functionalities. More information can be found on the WWW site of the project[11].

Acknowledgments

The PicoLibre platform, which is itself hosted on a PicoLibre installation at ENST de Bretagne, demonstrates a successful cooperative work, integrating direct and indirect contributions from tenths of people. We want to thank them all, especially our collegues and students from the Groupe des Écoles des Télécommunications.

REFERENCES

[1] Free Software Foundation, "The Free Software definition," `http://www.gnu.org/philosophy/free-sw.html`.

[2] "The SourceForge hosting platform," `http://sourceforge.net/`.

[3] "The PHP-GroupWare WWW site," `http://www.phpgroupware.org/`.

[4] IDEALX company, "A chrooted SSH CVS server," `http://www.idealx.org/en/doc/chrooted-ssh-cvs-server/`.

[5] OpenLDAP, "An open source implementation of the Lightweight Directory Access Protocol," `http://www.openldap.org/`.

[6] "The mailing-list manager Sympa," `http://listes.cru.fr/sympa/`.

[7] "PicoLibre platform hosted by ENST Bretagne," `http://picolibre.enst-bretagne.fr`.

[8] G. Ouvradou E. Cousin, "Valorisation de projets d'étudiants sous forme de logiciel libre," in *Pédagogie par projet dans l'enseignement supérieur : enjeux et perspectives*, Brest, France, June 2001, ENST Bretagne, pp. 29–36.

[9] "The CoopX project," `http://coopx.eu.org/`.

[10] Mohamed Wazni, "Protocoles d'échange entre serveurs coopératifs," M.S. thesis, École Nationale Supérieure des Télécommunications, June 2001.

[11] "PicoLibre WWW site," `http://www.picolibre.org`.