FREE SOFTWARE IN SUPERIOR EDUCATION: TALKING ABOUT FREEDOM TO FUTURE PROFESSIONALS

Willians Jorge Rodrigues (willians@igbt.sel.eesc.sc.usp.br)
Francisco José Monaco
Adilson Gonzaga
Manoel Luis de Aguiar

Departamento de Engenharia Elétrica – EESC – Universidade de São Paulo
Av. Trabalhador Sãocarlense, 400 – Centro
13566-590 São Carlos – SP

RESUMO

Este artigo reporta a experiência de uma iniciativa de disseminação de Software Livre em uma escola de tecnologia, destacando investigações sobre as preocupações de estudantes com relação à sua futura atuação profissional nesse campo.

ABSTRACT

This paper reports the experience of an initiative to disseminate Free Software in a school of technology and highlighting investigations on the students' concerns regarding their future actuation in this field.

1 FREE SOFTWARE IN A SCHOOL OF TECHNOLOGY

The GNUsp Project [1], leaded by an open group of students and researchers at the University of São Paulo (USP), is a volunteer initiative whose aim is to promote the Free Software (FS) movement within the University. This goal is currently seen after through the following main action lines:

- to promote the adoption of free software in both education and research activities at the University, by disseminating knowledge on its benefits and offering effective (both on-line and in loco) support to users over the campus;
- to encourage both students and researchers to adhere to the FS principles when working on software development within the University, either in academic or scientific projects, by maintaining an infrastructure of communication and mutual cooperation;
- to improve software developers' background on FS utilization and development, by offering a knowledge base and training tutorials at levels that range from basic software utilization to advance programming techniques.

From the three-year experience at USP campus in São Carlos, we have learned that providing our community with a suitable support base has been an essential stimulus to novice users. Likewise, in the development field, the interaction between experts and beginners (by means of the GNUsp facilities such as the mailing lists, Web database, software repository mirror, periodic meetings, advisory board etc.) has been highly effective in raising the overall programmers skills.

Among other actuation areas, FS believers involved in the GNUsp Project are committed to an orientation program devoted to introduce new users to the elementary FS paradigms, as well as giving them advice on the directions of the free software movement (FSM). In these times of Linux glory, with the open source software under the media's spotlight (specially the lay media), having a clear view of the ongoing revolution is very important – that's why GNUsp, rather than a LUG[1], is formally a GNU User Group (GUG?), since we base our work on the GNU software collection [2].

Nonetheless, while we have reasons to commemorate our success in enlarging the number of GNU users in our community, we have come across several difficulties along the way. Among them, we highlight those arisen from the fact that our activities are carried out within a school of engineering, where people are preparing to become well-succeeded professionals.

We have noticed that, in spite of the enthusiasm that the philosophy of freedom and cooperation raises in the students' youth spirit, these are subtly challenged by grave concerns which claim for attention when it's time to consider the future, the "real-life". The quotes are meaningful: they reflect the way many students do think about the problem when the time is near and they must decide what they will do next: "out there is the market, is the jungle; there is no time for dreams of freedom anymore" – with this belief in mind, many assume that the party is over and that it is time to work seriously.

Sadly, we have discovered that working seriously, in this case, means forgetting the student-like hopeful soul, dress-

[1] LUG: widespread acronym for "Linux User Group".

ing up in executive-fashioned way and waking up from the free software dream. Indeed, due to the pragmatic view of the professional world that many students of technological disciplines tend to acquire, this attitude is less rare than one might expect. This has constituted one of the obstacles to the dissemination of FS culture among young engineers and computer specialists, who end up by surrendering to the "inexorable hegemony of proprietary technology" when they have to make a choice in their professional career.

From exhaustive discussions on this matter, carried on by the light of the experience earned from the GNUsp activities, we have identified a few major points worth or being reported. By means of them, we aim to share with other FS researchers and practitioners the main concerns we were required to face in this front, as well as the arguments that have shown to be valuable to address them.

2 Free Software and Professional Actuation

When we organized the first seminar on Free Software under the auspices of the GNUsp Project, one of the first difficulties we came across was an unexpected embarrassment regarding the discussion of its underlying paradigms. Admittedly, talking about moral issues and personal values amid the fairly impersonal atmosphere that pervades the environment of a school of engineering was not an easy task, neither was the effort to attract the attention of students used to thinking in terms of commercial values and industrial trends.

It was clear to us, however, that the Free Software concept does concern to an inherently techno-social matter and that the attempt of dissociating both fundamentally complementary aspects to deliberately avoid polemics is a misleading attitude that breaks up the conceptual consistence of the original proposal. We have therefore accepted the challenge and decided to understand the psychosocial factors behind this hindrance.

2.1 How We Came to Where We Are

Back to the primordial days of the computer era in the late 60's and early 70's, the overall panorama regarding software utilization was noticeably diverse from the one existing today. By that time, the restrict number of software users could be accounted not only to the costs of computer hardware, but also to the operational complexity of rudimentary systems. As consequence, a set of peculiarities characterizes this period. It was marked by the predominance of the *user-driven in-field development*, i.e. once there was not many available off-the-shelf commercial packages, users were usually required to develop their own software, within the environment where it would be utilized, be it in industry, research centers or schools.

There was nothing unusual with respect to sharing code in that time, since most users were interested only in the use they could make from the software, instead of on the profit they could obtain by selling commercial packages. Thus, the *open source* model and the use of *non-proprietary* technology and standards were the natural choice in order to allow all interested users to take part in the worldwide collaborative effort.

However, during the transition from the 70's to 80's, the lowering of hardware costs and the simplification of user interfaces which accompanies the mainframe-to-PC migration has opened the doors to todays dissemination of computer utilization among non-specialist users. Soon, private companies become aware of the huge potential market, which was been formed, and it was then that software began to be treated as merchandise.

Adopting the *proprietary* approach, software suppliers intended that many users bought their products, thereby they have made use of legal and technical artifices to forbid both non-authorized copies and free shared utilization. Likewise, once former collaborator developers are then seen as potential *competitors*, not open but *closed source* began to be used in order to prevent copyrighted code from being further improved and adapted by others.

In-field continuous optimization yields to adoption of closed packages and users were then withdrawn from development process, which became *manufacturer-driven* and thus subject not only to the customers needs but also to the commercial strategy of the technology holders. These, in turn, replace the idea of a collection of common resources, which could be cooperatively improved with the market conquest through the imposition of proprietary standards.

2.2 From the Community to the Market

It has been reported by software professionals that this subtle paradigm shift compelled them to change from the usual open collaborative work to the closed competitive scheme. In order to fit into the new business model, knowledge sharing which was intensively practiced through the Internet was then replaced by the new orientation towards information hiding, while the freedom to cooperate with colleagues was substituted by the isolation of small adversary teams. As result, the instinctive ideal of working for the community was lessened in favour of individualist objectives and the volunteer spirit, which had been so expressive among programmers until them, was finally rendered to reminiscences of an ancient time.

In this scenario, the FS movement may be seen as natural reaction of the former software community. Briefly ex-

plained, its approach is meant to be more suitable for both users and developers not only because is enforces the convergence of collaborative efforts in the development of hight-quality cost-effective software resources, but also because it enables an alternative professional attitude which FS proponents believe to be more tuned to the human values lauder by broadly recognized conceptions towards a more fair and prosperous society.

Thus, if it is definitely queer to conceive anyone arguing against freedom and cooperation, why it so difficult to talk about it in the sphere of professional education? Actually, the answer becomes apparent once we realize how this question has been effectively concealed by means of ideological artifices.

We begun by noticing that early during their life in the college, technology students are taught to think about their professional careers, what is frequently stated in terms of "being prepared for the marked", actually in such an extent that, for many, the word *market* ends up by turning into the holy grail of a prospective worker. Not that being concerned about the field for professional actuation is not necessary, but the way the problem is presented and the relative importance of values such as economical viability, personal realization and ethical issues are not always balanced accordingly.

As a matter of fact, the market is the battlefield, where the weak is subjugated by the strong. Mainly in the technological areas, where human-related studies are less frequent than in social disciplines, the golden rule is *being competitive*. To think fast, to act first, to fight, to survive – hence the lemma "to survive, you must win; to win, you must be the best". Products, companies, professionals, everything, everyone is supposed to be competitive; in this world, not being competitive is being weak, fated to be a looser. Actually, this war-like atmosphere has gone so far that, beyond competition, some now talk about "assuming an aggressive campaign against adversaries", lending the well-succeed profession the appearance of a predator amid wild enemies, stimulating students to assume this posture as a noble personal character.

Competition, though, is opposite to cooperation. That's why Free Software concept, based on volunteer spirit and sense of community seems so odd-fashioned in this field nowadays. After all, in this arena where one's goal should be to defeat his potential competitors, there is no place for the dreamers; concerns about moral and ethical principles such as those implied in the FS proposal just do not apply. The ideological artifice worked and many professionals were convinced that the proprietary software is a natural evolution from an early rudimentary "ingenuous" activity into a structured business branch.

2.3 The Hacker Counter-proposal: Back to the Community

FS proponents, conversely, believe that it just deals with the replacement of one possible model with another one, which is not necessarily better. From the several approaches we have experimented in order to void those arguments, the rescue of personal values has proven to be the most effective. Indeed, this is dormant flame that, when evoked, is the door to bring back our colleagues to the "real real-life".

We remind them that, since proprietary software became dominant and the in-field development was replaced by the use of off-the-shelf closed packages, most developers have actually been using sophisticated applications rather than dealing with raw code and computer internals. Therefore, even within professional instruction programs and universitary courses, students feel unmotivated to learn the basics of computer engineering; this is left to a restrict set of large world-leader companies, while programmers have been continuously pushed towards what some recognize to be a generation of well-trained users, strictly skilled in one or other proprietary technology. This clue can be found in the legendary quote: "Do you pine for the old days of Minix-1.1 when men were men and wrote their own device drivers?" [3]. The words by the young Tourvalds materialize the feeling, which motivates it. It values the talent and personal commitment in opposition to the professional impersonalization that proprietary software approach has imposed by subjugating the open cooperative community. It also criticizes the devaluation of technical competence and individual merit, rescuing the social importance of the volunteer spirit and the sense of community.

Naturally, such initiative revels the complementary perspective of the Free Software phenomenon, that the Community comprises of people committed to human values and, as it is expectable, they prize personal qualities such as fidelity to own conceptions, idealism and open social relationship. Thus, the pronounced personality exposure and the indignation face to the violation of ethics and moral principles, even in the name of the marke, are indeed features of such a "hacker culture". In fact, these features explain how this literally anarchical organization can bring together so many different people – from teenager and undergraduate students to PhD researchers, programming experts and skilled engineers – and have them working interactively and coherently.

Finally, the essential argument against skepticism: good examples; fortunately, the Free Software Community has made them abundant.

REFERENCES

[1] GNUsp. The gnusp project. GNUsp, 2000. http://www.gnusp.br.

[2] GNU. The gnu project, 1985. http://www.gnu.org.

[3] Linus Tourvalds. Initial announcement of Linux project. posted to the USENET in alt.os.minix, 1991.