DID YOU SAY COMMERCIAL SOFTWARE?

CONTROVERSIES AROUND THE MEANING OF A WORD

Stefano De Paoli*, Maurizio Teli**, Vincenzo D'Andrea***

* Sociology Department, National University of Ireland Maynooth (Rep. Of Ireland),

Stefano.Depaoli@nuim.ie

** Faculty of Sociology, University of Trento (Italy),

maurizio@maurizioteli.eu

*** Department of Engineering and Computer Science, University of Trento (Italy),

vincenzo.dandrea@unitn.it

ABSTRACT

One of the suggested topic of the "3rd FLOSS International Workshop on Free/Libre Open Source Software" is "Competition between FLOSS and commercial software", stating in this way a clear and defined separation between the words "FLOSS" and "commercial software". In this paper we challenge the use of the word commercial as an antonym of FLOSS, showing how the commercial character of FLOSS is shaped by social practices embedded in the daily activities of software development communities. The mistaken separation "FLOSS vs. commercial" can be overcome looking at the practices of the software industry at large, practices that mixes different gradients of FLOSS and commercial activities everyday. In particular, we propose examples taken from two case studies: the Geographical Information System known as GRASS and the Operating System known as OpenSolaris. GRASS is a system covered by the GNU/GPL software license and developed by a community of volunteers. OpenSolaris is instead backed by one of the major IT world player Sun Microsystem.

In conclusion, in this paper we unfold how FLOSS developers, both volunteers and corporate employees, built the meaning of the word commercial as it relates to their daily practices and to what they produce. Indeed, we hope to provoke reflective practices among FLOSS scholars and policy makers, deepening the understanding of the relationships between FLOSS and commercial practices, discarding the use of "commercial software" as a synonym of "proprietary software".

PRELUDE

Professor: Have you decided the topic of your dissertation?
Ph.D. Student: Yes, I think I will focus on the commercial aspects of free software.
Professor: Mmm... interesting, but if it's free, how can it be commercial? Free and Open Source software are against commercialization. This is all about gift economy, think about that.

[fictional conversation between a Ph.D. Student and a Professor]

First Developers: I think there are a lot of people out there who would love to make maps but they can't afford commercial GIS programs.....

Second Developer: You probably speak about "proprietary" or non-free GIS software.
[true conversation between developers in a Mailing list[1]]

First Developer: [these]....search keys in Google give a number of commercial programmes for converting SOSI to shape and other formats,

Second Developer: I guess that you've meant proprietary converters. There are many publications where I saw the confusion between commercial programs (standing for development as revenue generating activity and sometimes for quality) and proprietary programs (not freedom attached, you might need to pay for licenses)

[another true conversation between developers in a Mailing list[2]]

1 From http://n2.nabble.com/-GRASSLIST%3A1013--creating-a-desktop-GIS-application-using-GRASS-td1854674.html
2 From http://n2.nabble.com/-GRASSLIST%3A2879--SOSI-files-td1850336.html#a1850338

1. COMMERCIAL OR PROPRIETARY? THIS IS THE PROBLEM.....

Almost everyone teaching and researching on FLOSS has been involved in a conversation such as the fictional one we used to begin this paper. Indeed, the academic debate has often contrasted Free/Libre Open Source Software (hereafter FLOSS) to what is called "commercial software". Our goal with this paper is to shed new light on the use of the word *commercial software* as opposed to FLOSS, challenging the use of these terms as antonyms. We will show the process of co-construction between the commercial character of FLOSS and the social practices embedded in the daily activities of software development communities, as those exemplified in the true discussions among developers in the prelude.

In this paper we do not discuss concrete aspects of business related to FLOSS (i.e. the so called FLOSS business models, see for example Raymond 1999) nor we discuss what Perens (2005) has called the raise of the economic paradigm of FLOSS. We are also aware that influential contributors to the FLOSS debate, such as Lerner and Tirole (2005) or Kogut and Metiu (2001), have well pointed out that FLOSS communities may derive substantial commercial benefits from open source licensing and other practices such as consulting. In this work our interest goes on how the word "commercial software", is defined and used in everyday software development practices (Lin, 2005). In particular we are interested in how this word and its meaning are used as a way of making a strategic order and sense of the worlds inhabited by FLOSS stakeholders.

As it is widely acknowledged, FLOSS is a way of licensing software that gives to the users the ability to change and share the software (these are called freedoms of software by the Free Software movement – see FSF, 2004). However, FLOSS has been considered also as an innovative software development methodology in which a hierarchical and centralized development model (defined by Raymond to as "Cathedral model") is opposed to a flat and distributed model (typically known as Open Source or Bazaar). Generally speaking FLOSS has its grassroots in the idea that the software cannot be exclusively controlled by its producer. Rather the software has to be fully available for the community of users that has produced it. Therefore the definition of FLOSS can be grounded in a opposition with the so-called *proprietary or closed source software model*. The latter is a development model in which, usually, the producer bases the business on selling copies of the software (sells a license for using code binaries but not the source code) in exchange of money[3], whereas the

3 It is worth mention that the software known as Freeware is proprietary software distributed gratis, usually for a limited period of time (trial period).

software is developed, in full secret, only by the producer's employees.

In this regard, in many, and even influential, cases the term *commercial software* has been assumed as synonym of what we have just described as *proprietary software.* It is interesting to observe that even one of the spokesperson of FLOSS, Eric Raymond himself, took this opposition for granted in his famous essay *The Cathedral and the Bazaar* (1999b):

> "Perhaps in the end the open-source culture will triumph not because cooperation is morally right or software "hoarding" is morally wrong (assuming you believe the latter, which neither Linus nor I do), but simply because *the commercial world cannot win an evolutionary arms race with open-source communities* that can put orders of magnitude more skilled time into a problem ". [italic emphasis added]

As we can see Raymond opposes what he calls open-source to a not better defined "commercial world", arguing that the latter cannot compete with the evolutionary power of FLOSS communities. By reading statements like this, readers have no reason to doubt that there is really a sharp opposition between *commercial software* and FLOSS. However, as we will see in the empirical part of this paper, this opposition between FLOSS and commercial software creates a discrepancy with the existing FLOSS developers rhetoric and the ways they use the world commercial in relation to FLOSS. Indeed from our empirical observations it appears a different story in which for FLOSS both the *nature* (whether FLOSS is commercial or not) and the *process* (how FLOSS is commercial) of being commercial is important.

The sharp opposition that sees FLOSS as the antonym to commercial software can be found in several influential literature contributions. The table 1 below summarizes some of them and how they set the FLOSS/commercial opposition, highlighting (by italic emphasis) the problem we are describing. It is also interesting to note that this problem relates to several different points of view ranging from historical accounts of major FLOSS events to licensing discussions.

It is our opinion that the main problem with literature is that it does not take in account the meaning of the word "commercial" for the actors involved in the FLOSS development. Indeed, we challenge the literature in this by arguing that FLOSS stakeholders use the word commercial to refer to FLOSS as part of well defined and concrete strategies used to affirm the peculiar character of FLOSS. By contrast, we think that in the literature the word commercial is somehow taken for granted as the antonym to FLOSS and this seems to be more than a simple misunderstanding. Rather, we think that this is both a short-cut for defining a phenomenon (the closed, proprietary software) by assigning to it characteristics that clearly can be extended also the supposed antonym, as well as it is a very imprecise way of using a word that does not take into account the meaning assigned to it by the FLOSS actors themselves. In this sense we agree with Wheleer that has clearly pointed out that the opposition between FLOSS and *commercial software* is not only imprecise, but also mistaken for several reasons:

> (1) the rise in commercial development and support for FLOSS, (2) most FLOSS projects' goal to incorporate improvements (which are actually a form of financial gain), (3) official definitions of "commercial item" that include FLOSS, and (4) FLOSS licenses and projects that clearly approve of commercial support. Terms like "proprietary software" or "closed source" are plausible antonyms of FLOSS, but "commercial" is absurd as an antonym. " (from Wheeler, 2006).

We think that these reflections proposed by Wheleer point in the right directions and requires a serious empirical investigation aimed at highlighting the everyday use of the word commercial in relation to FLOSS. In doing so, we propose examples taken from two different case studies, that provide us with a good degree of varieties: (1) the Geographical Information System known as GRASS[4], a system covered by the GNU/GPL software license and developed by a community of volunteers and (2) the Operating System known as OpenSolaris[5] backed by Sun Microsystem, that is one of the major IT world player.

The paper is organized as follows: we initially describe our approach; then we move on describing the empirical cases of GRASS and OpenSolaris; we have then a discussion and the conclusion.

Table 1: FLOSS and Commercial Software used as antonym.

<u>Historical viewpoint:</u>
A good way to get a first grasp of open source software is to observe how, throughout its history, *it has differed from commercial software*. (Von Hippel and Von Krogh, 2003a, Online version) I briefly discuss the case of *Linux entering the markets for server operating systems previously dominated by commercial software enterprises*. (Bitzer, 2004, Online Version)
● **<u>Licenses viewpoint:</u>**
Open Source Software is given away for free by the developers who write it, both in the sense that it is provided at a nominal charge and that *it is licensed to users without the legal restrictions typical of commercial software.* (Healy and Schussman, 2003, p. 2) However, the fact that open source software is freely accessible to all has created *some typical open source soft-*

4 http://grass.osgeo.org/

5 http://opensolaris.org/os/

ware development practices that differ greatly from commercial software development models—and that look very much like the “hacker culture” behaviors described earlier (Von Hippel and Von Krogh, 2003b, p. 211)

Transactions among agents in an open source environment are regulated by a variety of licence agreements which, in different ways and degrees, *protect the openness of the source code and prevent the commercialization of cooperatively developed software*. (Lanzara & Morner, 2005, p. 86)

<u>Development Model viewpoint:</u>

The open source development (OSD) model is different from traditional in-house commercial development processes in several fundamental ways. First, the usual goal of an open source project is to create a system that is useful or interesting to those who are working on it, not to fill a commercial void. Developers are often unpaid volunteers who contribute towards the project as a hobby; in return, they receive peer recognition and whatever personal satisfaction their efforts bring to them.(Godfrey and Tu, 2000, p. 132)

On the other hand, *a major difference from commercial software development is that, in open source projects, the requirements are not fixed over the life time of the software*. According to the requests of programmers and especially users, new functionality is added. This violates the assumptions of most traditional models for software development effort estimation. (Koch S. and Schneider, 2002, Online version)

If we look at *the amount of code produced by the top Apache developers versus the top developers in the commercial projects*, the Apache core developers appear to be very productive, given that Apache is a voluntary, part-time activity and the relatively “lean” code of Apache. (Mockus et al., 2002, p. 324)
[…]
In the “free” world of OSS, patches can be made available to all customers nearly as soon as they are made. *In commercial developments, by contrast, patches are generally bundled into new releases*, and made available according to some predetermined schedule. (Mockus et al., 2002, p. 330)

<u>Source Code Access viewpoint:</u>

From an economic point of view Open Source software can be analysed as a *process innovation*: a new and revolutionary process of producing software based on *unconstrained access to source code as opposed to the traditional closed and property-based approach of the commercial world*. (Bonaccorsi and Rossi, 2003, Online Version)

Later, *when commercial software development increased and often only the software vendor had access to the source code of a program, OSS became an attractive alternative* since it enabled the users to adapt and improve the software according to their personal needs. (Hertel, et al., 2003, Online Version)

Most commercial software is released in machine language or what are called "binaries" — a long string of ones and zeros that a computer can read and execute (Weber, 2004, p. 4).

Efficiency view point:

On first examination, open source software seems paradoxical. Open source software is a public good provided by volunteers—the "source code" used to generate the programs is freely available, hence "open source." Networks of thousands of volunteers have developed widely used products such as the GNU/Linux operating system and the Apache web server. Moreover, *these are highly complex products and they are, arguably, of better quality than competing commercial products*, suggesting that open source provision may be highly efficient. (Bessen, 2005, p. 1)

Perhaps in the end the open-source culture will triumph not because cooperation is morally right or software "hoarding" is morally wrong (assuming you believe the latter, which neither Linus nor I do), but simply because *the commercial world cannot win an evolutionary arms race with open-source communities* that can put orders of magnitude more skilled time into a problem. (Raymond, 1999b, Online Version)

2. THEORY APPROACH

Before approaching the empirical case studies, we would like to spend a few words on the theoretical approach used in this investigation. The research presented here comes from two doctoral dissertations in which we have adopted an emergent approach to account for the socio-technical process related to FLOSS development, investigating the licensing dynamics as they relate to power (De Paoli, 2008) and the enactment of freedom practices from socio-tehcnical systems (Teli, 2008). In particular by saying that we have adopted an emergent approach we mean that we have emphasized the accounts that are directly provided by the actors we are studying: in the specific case of this paper how the developers, users or even companies (but this is true for all the stakholders) involved in FLOSS development account for their way of seeing the commercial nature of software.

Therefore we do not have a scientific theory or a battery of hypothesis to test but, on the contrary, we provide a description of how the actors themselves make sense of their world. In doing so we follow an important tradition of research related to phenomenology (see in particular Garfinkel, 1967) as well as an approach known as Actor-Network Theory (ANT) (see in particular Latour 1987; 2005). Methodologically speaking, following this tradition of research means that it is not the duty of the observer to decide in advance the attributions of social and technical elements (often referred to as entities, Callon, 1986; or actants, Latour, 1987) of the technological systems. To use Garfinkel own words (1967), these, instead, are "*Ethno*"-"*methods*" that emerge from the negotiations surrounding the System itself: they are practical and situated actions (*methods*) that are not made in a separate world by a natural observer, because the social actors (*Ethno*) are already being-in-the-world (Heidegger, 1927; Ciborra, 2002). Our idea therefore is that the commercial (or non commercial) nature of FLOSS should be seen as an Ethno-method, something that people use in order to make order and sense in the socio-technical world in which they live and work.

One of the key elements for approaching this type of situated practices and their social and technical attributes is to focus on the moments of rupture. For example, Akrich (1992) observed that the black-box nature of technology prevents us from observing the negotiations between designers and users. In this light for Akrich we need to focus on disputes around technology or situations in which devices go wrong, as the crucial moments that reveal social and technological elements that are often taken for granted. Similarly, Winograd and Flores (1987) proposed to focus on the situations of technological breakdown. In breakdown situations we can understand how what we often take for granted is problematic, and how this rupture leads to new and unexpected innovations. Properties of things, as Winograd and Flores points out, are not inherent in the world but often arise in the event of something breaking down. In this paper we propose therefore to study the enactment of the commercial nature of FLOSS as a situated practice, that reveals its characteristics in the disputes or breakdowns around technological and social aspects of FLOSS practices and technologies themselves.

In the light of the idea of focusing on the "rupture" moments, we think it is important to draw attention to the mailing lists (ML) discussions of FLOSS projects. MLs are, in a broader sense, the computer mediated spaces where we can follow the discussion between stakeholders involved in FLOSS development as they relate to rupture moments in technology[6].

6 In the case of GRASS, data has been gathered using an archaeological approach (De Paoli, 2009) which focuses on the centrality of archives of mailing lists and source code. In the case of Opensolaris, data has been collected through a twenty months cyberethnography (Teli et al., 2007) starting from the OpenSolaris.org website, following mailing lists, blogs, IRC channels, news, and interviewing developers.

3. WHAT DOES IT MEAN COMMERCIAL?

In this paragraph we will analyse closely the relationships between FLOSS and the word *commercial* as they emerge directly from software developers practices. In particular, we focus on a controversy around the definition of commercial software related to the GRASS project.

GRASS is an interesting case for studying the controversies around software licensing. The system was originally developed by the US Army (1980-1996) in the Public Domain. In 1997 however a new development team (known as the GRASS Development Team, or simply GDT) took over the development and in October 1999 decided to release the system under the GPL V. 2.0.

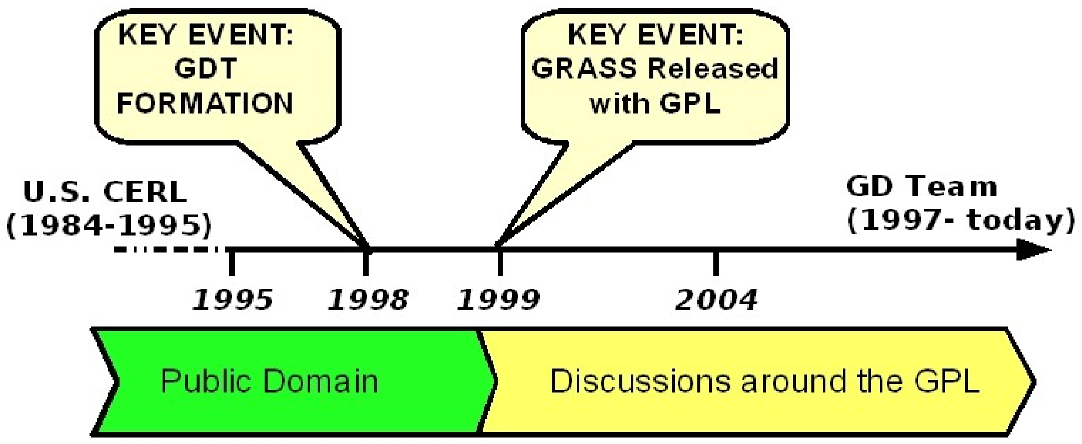

Figure 1 – Key events of GRASS case study

The license change in the GRASS case study constitutes one of those moments of rupture that we have identified as crucial for observing elements that are often taken for granted. Indeed, the fact the GPL was not the original license of the system has triggered an ongoing process of conflicts resolution between the GPL itself and the licenses of several commands/modules/libraries of the GRASS system. Whenever the compliance with the GPL has emerged as problematic, this has also constituted a rupture moment in which we can observe interesting dynamics between developers that would otherwise lie unobserved in the background.

3.1 HOW THE GPL DEFINES THE "COMMERCIAL" NATURE OF FREE SOFTWARE

In this paragraph we will briefly account for the relationships between the software and its commercial nature as it is defined by the terms of the GNU General Public License (GPL) in the version 2.0. This short digression on the GPL V 2.0 is important in order to follow the case of GRASS. In particular it is worth mention that we focus on version 2 of the GPL because it directly relates with the events we will discuss later.

In a previous work (De Paoli et al., 2008) we have described how the terms of the GPL constitute what in the Actor-Network Theory (Latour, 2005) vocabulary is often referred to as a group of "*trials of strength*". With this concept it is meant that a textual statement or a technological arrangement has often the goal of anticipating other actions, considered dangerous for certain specific strategies. In the specific case of the GPL 2.0 this is clear for example with term 2b[7], often referred to as Copyleft. One of the goals of the Copyleft term is to anticipate all the attempts to turn Free Software into proprietary software by creating closed source derivative works. This is a "trial of strength" in the sense that the Copyleft term fulfils the goal of anticipating those trials that can lead to a failure. Indeed, the possibility to release derivative works as proprietary software is a trial of strength that would lead to a failure of the overall Free Software. In this light the definition of Free Software as a particular type of commercial software posses the same dynamic that exists for the the Copyleft term. In this case however, another term of the GPL 2.0 is fundamental, the term 1. [8] :

> **1.**You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, [...]
>
> You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

Literally, this term states that the user who receives a copy of a GPL covered software is free to distribute copies of such software. What is more important is that this distribution can be done even in exchange of a fee, as long as all the freedom of software are kept intact (GPL, Preamble). This is a first important provision of the commercial nature of GPL covered software. Moreover, still according to the term 1., it is possible for a company to provide warranty protection to the software in exchange for a fee. In fact, every software[9] is usually distributed "as is" without any warranty on it. However a company may introduce a warranty protection asking the customers to pay a fee in ex-

7 In the GPL V. 3.0 the "Copyleft" principle is contained in term 5.

8 In the GPL v.3.0 this term is the number 4. and says "You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee. "

9 This is true for both proprietary and Free and Open Source Software.

change. The term 1. of the GPL V 2.0 defines therefore Free Software as commercial, specifying some of its fundamental dimensions: the distribution of software and the extension of warranty. Both these dimensions clearly relates to commercial possibilities[10] that can be exploited by FLOSS users. Hence we have a relationship between Free Software and commercial software in the limits that the business cannot be based on restricting any copy of the software (i.e. proprietary software).

3.2 GRASS AS COMMERCIAL SOFTWARE

We begin now with a presentation of the GRASS case. Interestingly in GRASS discussions what is a commercial software is a matter of definition that is achieved via negotiations among developers. These negotiations are also based on an active role of licenses (especially the GPL) in setting an horizon of possibilities thorough trials of strength, between what is and what is not possible to do with a GPL covered software. The following message exemplifies this situation (emphasis added):

> "Searching for the keyword "commercial" I turned up a couple of problematic cases. [...]
>
> There are several commands by Author: [Name]
>
> Permission to use, copy, modify, and distribute this software and its documentation f*or* ***non-commercial purposes is hereby granted.*** This software is provided "as is" without express or implied warranty."
>
> [BR, GDML, 08 September 2000]

The above message is taken from the GRASS users Mailing list. In particular, a developer posted what seems to be the copyright statement of a software module contained in the pre-GPL GRASS code. While the statement clearly inherits the freedom to copy and modify the software (*Permission to use, copy, modify, and distribute this software*), it also clearly states that the software commands that it covers cannot be used for any commercial purposes (in bold). This situation is described by the developer as "problematic". Indeed, GRASS developers recognized here an incompatibility between this copyright statement that denies the use of software for commercial purposes and the GPL itself. Here we already witness a clear relationship between the word commercial and the GPL'ed covered software.

In this case the conflict between licenses was easily solved by asking the module developer to modify the above copyright statement. In particular he agreed to release his GRASS commands un-

10 This does not exclude other "business" with Free Software such as for example the possibility for a programmer of getting paid for the act of modifying the software, or compiling and selling software manuals and so on (see this for various business models and free software, Perens, 2005).

der the same GPL terms. Easy solutions to licensing controversies are however more the exception than the rule. Indeed, the meaning of the "commercial" nature of FLOSS can better be grasped with a more concrete and controversial example, in which a major breakdown has revealed important elements for our enquiry.

File formats are ways of organizing computer data. A common existing issues with data formats is the existence of both closed and open format. In the former case the specification of how the data are organized is kept secret by the producer while, in the case of open data format, the specifications are fully published[11]. GIS technologies use a varieties of Geographical data formats and for this reason import-export functionalities are fundamental. In the remaining of the paragraph we are going to describe the conflict between the GPL and the license of a well known import-export library used for managing the data format known as DWG: the library OpenDWG, a piece of software that it is distributed in both binary and source code form. This library has been written with the goal of providing a way for manipulating the DWG closed data format by a "*membership-based consortium of software companies, developers and users committed to promoting the open exchange of CAD data now and in the future* " (from http://www.opendwg.org/). This consortium is called Open Design Alliance.

This library (OpenDWG) was introduced in GRASS with the development of a brand new GRASS vector library in 2003. In a way, the OpenDWG library enabled the GRASS users to use DWG maps, in particular thorough a specific GRASS DWG import command, known as *v.in.dwg*. Some time after the introduction of this library in the GRASS framework however, the following message was posted on the GRASS Developers Mailing List by a GRASS developer:

> Noticed that v.in.dwg from GRASS 5.1 [...]
>
> uses the proprietory library opendwg. As I believe that it also needs the GRASS libraries which are under GNU GPL, this means that v.in.dwg has a severe license problem.
>
> [BR, GDML, 13 May 2003]

In this message the Library OpenDWG is described as having serious licensing problems. The problem is that OpenDWG and GRASS were becoming a single software at compilation time and hence a single derivative work, figuring in this way a violation of both the GPL and the OpenDWG license. The GDT took then the decision to eliminate the OpenDWG library from GRASS. The direct result of this, however, was that of making the GRASS system lacking any possibility to manip-

11 It is important to remark that the division between open and closed format does not mirror the division between Free and proprietary software. In fact many open data format are realised by proprietary software companies, such as for example the well known Adobe PDF.

ulate maps in the DWG data format. This is a clear case in which the freedom of software seems to be more important than the functionalities of software itself.

The debate around the OpenDWG inclusion in GRASS did not end here. Almost one year after the elimination of OpenDWG library, one of the GRASS developers posted the following message, describing the terms and conditions of the OpenDWG library:

> To my surprise, their web site description of Associate Member terms and conditions permitted the distribution of the libraries in software that is distributed free of charge. This certainly fits GRASS.
>
> [MB, GDML, 26 August 2004]

According to this developer the "status" of Associate Member of the OpenDWG Alliance grants the permission to use and distribute the library in software which are "free of charge", a situation that, still according to the developer, "fits GRASS". In order to be sure about the possibility to use GRASS the developer decided to call the OpenDWG Alliance head quarter, asking for clarifications about the use of their library:

> So I called them this morning. I had a good discussion with the membership coordinator with the Open Design Alliance. He assured me that the alliance's intent was only to restrict or control commercial use of their libraries, not use in educational or free software.
>
> [MB, GDML, 26 August 2004]

The phone call seemed to clarify any doubt about the possibility to use the OpenDWG with GRASSlibrary. The following is an excerpt from the OpenDWG terms of use for the Associate membership, that was posted on the GRASS Developers Mailing List, followed by a comment from the developer:

> \> *Dear New Associate Member :*
>
> \>
>
> \> *I have given you access to the DWG files according to the Associate Member Agreeent.*
>
> \> *We allow access to our libraries for research purposes and development of free or*
>
> \> *internally used software only.*
>
> Unless there is some catch to the GRASS GPL license that I am missing (quite possible, I suppose, given my lack of legal expertise), I think we can distribute openDWG libraries with GRASS as long as we don't sell GRASS commercially--something prohibited by the GPL license.
>
> [MB, GDML, 26 August 2004]

According to the words of this developer then, it seems to be possible to distribute GRASS and the

OpenDWG library as compiled software as long as GRASS is not sold commercially. This action, according to him, seems to be prefigured by the GPL license itself, due its provision of preventing the commercialization of software. However this understanding of the GPL license was soon contradicted by another developer:

> The GPL in no way prohibits commercial distribution of software (look at all the GNU/Linux Distributions that sell GPL'd software). Free in the sense of free software (and in the sense of the GPL), does not mean non-commercial, it means the freedom to access, modify and redistribute modified version of the source code. But you have every right to sell GPL'd software, including.
>
> [ML, GDML, 27 August 2004]

This message tries not only to clarify that the word "Free" as it relates to Free Software does not mean gratis. The intention of this message seems to be that of pushing a specific interpretation in which FLOSS is also necessarily commercial software. Moreover, the commercial nature of FLOSS is sustained by concrete examples: the GNU/Linux commercial distributions, that are quoted in this message. These distribution constitute part of the "hinterland" (Law, 2002) of the commercial nature of Free Software: a socio-technical assemblage of things (composed of both social and material elements) that concretely sustain a certain definition of reality. In other words the concrete existence of GNU/Linux commercial distributions sustain the definition of FLOSS as commercial software. We can easily understand that what is at stake in this discussion is exactly the definition of what is a commercial software. Commenting the content of the phone call between the GRASS developers and the OpenDWG Alliance another GRASS developer posted the following message:

> I would say that Mr. Dahlberg does not know the definition of Free Software and would not have inlcuded Free Software in his statement. I am pretty sure he meant proprietary gratis software (aka freeware).
>
> [JOW, GDML, 30 August 2004]

The developer pointed out that the spokesperson of OpenDWG (Mr. Dahlberg) meant to address what is known as freeware rather than Free Software, in other words a type of proprietary software which is distributed free of charge. In order to clarify this confusion of terms between commercial and proprietary software, the following statement (taken from the OpenDWG website) was then posted on the GRASS mailing list:

> Open Design Alliance members have created the following free utilities, based on the OpenDWG Libraries, for your unrestricted, non-commercial use. Please note that inclusion of any utility in a commercial product does require commercial licensing[12]

This post finally clarifies that it is not possible to use the OpenDWG Library together with

12 See http://www.opendesign.com/downloads/guest.htm

GRASS, due to the commercial nature of GRASS itself which is granted by the terms of the GPL. Indeed the OpenDWG can be freely used, as it is clearly stated above, for non-commercial purposes.

Above all, the idea that FLOSS is not just the opposite of commercial software is part of clear strategy of developers and not just something that is part of academic critique and discussion. Several others examples can be taken from the GRASS case to justify this statement. Some remarks on this case study are however useful at this stage. In doing so, we can draw on the following message:

> Hello,
> as part of our marketing strategies as OSGeo we try to be careful in our wording. One trap that we try to avoid is opposing Open Source software to "commercial software" as this is not the appropriate antipode to what we are trying to say:
> http://en.wikipedia.org/wiki/Commercial_software
>
> The term "commercial" itself can be perfectly applied to Open Source and Free Software:
> http://wiki.osgeo.org/index.php/Commercial_Services
> http://wiki.osgeo.org/index.php/%22Commercial_Software%22
>
> The opposite to Free Software licensing is proprietary licensing and the opposite to Open Source development methodology is closed source. The distinction here is best formulated as Open Source vs. Closed Source (development wise) and Free Software vs. proprietary (licensing wise).
>
> From
> http://n2.nabble.com/UN%27s-program..%3A-ESRI-and-cities-mapping-td1879544.html#a1879547

This message makes even more clear that for developers there is a need for clarification about the precise meaning of the term commercial. In particular OSGeo is the recently founded Open Source Geospatial foundation (www.osego.org), an umbrella foundation that gathers several FLOSS project that have in common their geospatial nature. GRASS in this regard is one of the founding projects. As we can see OSGeo, in the definition of a clear strategy, is willing to promote the use of Geospatial FLOSS. In this strategy the use of words is taken as a serious matter. According to OSGeo indeed the commercial nature of FLOSS not only should be assumed as appropriate but will need to be considered an an inherent characteristic of FLOSS. In the above message it is clearly remarked that the term that characterize the antonym of FLOSS is proprietary software and that by contrast the term commercial can be perfectly used with FLOSS.

4. COMMUNITY BUILDING & COMMERCIAL SOFTWARE

If the word *commercial software* has a negotiated meaning emerging from everyday practices in voluntary based FLOSS projects, the involvement of corporations into the FLOSS panorama makes possible to think about the relationships between the emerging practices of developers and the strategic commercial plan of corporations.

In the case of OpenSolaris, the commercial side of FLOSS is performed through two different arenas of action: first, an OpenSolaris-based distribution that Sun provides to its customers; second, through the translation of the pre-existing proprietary software development, influenced by commercial aims, into a community based project. In both these arenas, the intersection between the commercial nature of Sun and the FLOSS development strategy, raises controversies and negotiations on the role of the corporation, for which the commercial nature of FLOSS seems to be taken for granted.

The fact that OpenSolaris has a commercial side is stated clearly in the first version of the document called "*OpenSolaris Governance Proposal*"[13] (later on called "*Constitution*"[14]), that in its first sentence defined OpenSolaris as "*an organization dedicated to the collaborative production of open source software for a family of commercial-grade operating systems.* " (OpenSolaris Governance Proposal (Constitution), Draft 00).

In this case, the definition of "commercial-grade operating systems", later on discharged, was used to identify a degree of quality of technology, as shown in the following part of the "Principles" of OpenSolaris: "*Quality is always a top priority. The OpenSolaris project will continue the long tradition of quality engineering established by the Solaris Operating System (OS).*" This concept of "quality" has been translated into specific development and distribution practices that are deeply affected by the intersection of the commercial interest of Sun Microsystems, by the possibility to enrol other commercial entities to participate, and by the previously existing development practices aimed at satisfying those conception of quality and grade of technology.

Siding that, Sun Microsystems strategy of releasing software technology under FLOSS licenses was connected to a specific rhetoric related to both the business and the software industry scenario. This rhetoric, is very clearly presented in a book written by two Sun engineers, who state in the very first lines of their work that: "*business is changing after the expansive thinking of the late 1990s followed by the lessons learned in the early 2000s: It no longer makes sense for every company to*

13 Retrieved at http://mail.opensolaris.org/pipermail/cab-discuss/2005-July/000763.html [Jun 10th, 2009]

14 Retrieved at http://www.opensolaris.org/os/community/ogb/governance/[Jun 10th, 2009]

make and own every aspects of its business" ((Goldman e Gabriel, 2005)). In this way Sun engineers underline how FLOSS can be conceived as a fundamental part of the way through which conduct business in the contemporary software market. The authors articulate their vision of "making FLOSS a business practice" as a "vision of community building" that creates fundamental connections between a specific aim and a set of artefacts designed to increase the participation of entities (including both stakeholders and other software), different from Sun Microsystems, its employees, and its technology. Specifically, here it is possible to highlight how software licenses and the infrastructure that has been built to support the development, integrate the vision of FLOSS as a business strategy and thereof as commercial software.

The license Sun Microsystems did chose to protect the OpenSolaris codebase was a brand new license, the *Common Development and Distribution License* (often referred to as CDDL), written with the aim of building a network of entities around OpenSolaris that is different from those that are enacted by other licenses such as for example the GPL (see De Paoli et al., 2008). For us, the most important thing is that Sun has chosena file-based license (on the model of the Mozilla Public License), with the aim of "*enabling creation of larger works for commercial purposes*"[15]. In other words, the aim of Sun is to allow the distribution of OpenSolaris code, still protected by the CDDL, with proprietary code in a software distribution sold for a fee. The most notable example is the same Solaris commercial version, distributed together with Sun's hardware or via a website (without support), and regulated by a traditional proprietary software Software License Agreement[16].

The license is the focus for understanding how the commercial side of FLOSS is discussed among developers, and especially for understanding how the behaviour of a commercial entity is at the intersections of different views of legitimate participation and benefits for a FLOSS project. Here, we refer to the integration of two OpenSolaris technologies, the tool *DTrace* and the filesystem *ZFS.* They has been included, thanks to the CDDL, in the Apple MacOSX Leopard operating system. To a certain extent, this case represent a moment of rupture in which interesting elements of the commercial nature of FLOSS have emerged. Indeed, this case was debated in the OSOL community in relation to the possibility of dual licensing OpenSolaris under CDDL and the GNU General Public License version 3:

> "Besides, another one of the SUN folks posted earlier today
>
> about how valuable people at Apple were because of the know-

15 CDDL FAQ, Retrieved at http://openmediacommons.org/CDDL_FAQs.html [Jun 10th, 2009]

16 http://www.sun.com/software/solaris/licensing/sla.xml

> ledge they shared about DTrace. You will have no disagreement with me that what SUN has done is incredible and wonderful. They have definitely done far more than any other company has, in my personal opinion, for open source.”
>
> [SW, osol-discuss, 31 Jan 2007]
>
> “Apple was mentioned as a point for the CDDL, as they supposedly wouldn't have used dtrace or zfs otherwise. Well, Apple has tended to do whatever it takes to use the technology that's available, and that has included using GPL'd software. If they're telling you that they would forsake the advantages of dtrace and zfs if it's GPL'd, then they're lying. [...]
>
> And finally, I'm not sure what you're expecting, but Apple has a terrible history of contributing anything to open source projects. So... why do you care about Apple?”
>
> [MC, osol-discuss, 2 Feb 2007]

In these two posts, we can clearly see how the fact of OpenSolaris technologies being adopted by a corporation in its commercial activities, being that Sun or Apple, is not under discussion, but what is under negotiation is rather “how” to be commercial. The behaviour of Sun is judged positively, because of the effort done to promote FLOSS at large, despite the construction of a proprietary product based upon OpenSolaris. On the contrary Apple's behaviour is disputed. Apple is described as not contributing to the development of FLOSS, but only as, to quote another developer “s*ucking the living daylights out of the open source community and putting nothing back* ” [AD, osol-discuss, 31 Jan 2007]. Therefore, license debates in the OpenSolaris case show how the commercial nature of FLOSS projects is not an issue as such, related to “*if FLOSS can be commercial*”, but what is at stake is instead “*how FLOSS can be commercial*”, also in opposition with proprietary development.

In relation to this, the OpenSolaris case provides us another interesting issue: the role of the technological infrastructure used to support the work of FLOSS developers in their everyday activity. Here we have at least two different points of attention: (1) the use for a long period of time of the Source Control Management TeamWare, used internally by Sun employees before the “open sourcing” of Solaris into OpenSolaris; (2) the “freedom of constraints” statements and its relation to technical limitation to the inclusion of new code into the main code base.

When OpenSolaris was opened to the public, the Source Control Management (SCM) used was TeamWare, the one used by Sun employees for the development of the Solaris operating system in its proprietary version. From the point of view of the ordinary practices of FLOSS developers, this technology and the organization of development was affecting the way through which an external developer could submit changes, updates, and patches. Indeed there was a necessary passage through Sun employees, who were the only ones entitled to submit changes to the OpenSolaris code. When this issue was brought to the attention of the OSOL community this was another of those moments of rupture in which development practices taken for granted by Sun employees became questioned. The mailing list debates were often pointing out how this kind of conduct (i.e. the employees acting as gateway) was not acceptable in a FLOSS community. For this reason the OpenSolaris Governing Board started a process of migration to a new SCM, known as Mercurial, with access granted to anyone who was able to gain the status of Contributor[17].

The so-called "Freedom of Constraints" is more relevant in relation to the fact that commercial practices, in a case of "open sourcing" like OpenSolaris, affect the ways through which FLOSS development can be done. The "Developer Reference" indicates that OpenSolaris - that will be the base for the new commercial version of Solaris (the forthcoming Solaris 11) - needs to adhere to retro-compatibility constraints, in order to allow the commercialization of Solaris with the same standard as before: that is, the possibility for Solaris customers to run applications developed for version 7 (or later ones) in every following version. This situation has been sometimes taken as a limitation:

> "One of the issues with ksh is making sure we don't break compatibility with older versions. But that's not an insurmountable barrier.
>
> \>
>
> \> Do we have to? How much of a clean sheet is OpenSolaris?
>
> It's not intended to be a clean sheet.

17 Contributor is defined in the CDDL as "each individual or entity that creates or contributes to the creation of Modifications" (http://opensolaris.org/os/licensing/cddllicense.txt), and by the Constitution as "A participant who has been acknowledged by one or more Community Groups as having substantively contributed toward accomplishing the tasks of that Community Group, or by the OGB for at-large contributions, shall be termed an OpenSolaris Contributor. "(http://opensolaris.org/os/community/ogb/governance/).

> But I'm with you on wanting to move the shells forward (probably reclassifying to "Standard" to properly set expectations for future incompatible changes).
>
> > One of my Linux advocates is always complaining about all the backwards compatibility baggage Solaris carries. I tend to agree with him on many occasions. Note the bash in 10 breaks backwards compatibility by not exporting HOSTTYPE - so a precedent has bee set!
>
> The overriding rule is "document the stability level of interfaces".
>
> [SO, code, 5 Aprile 2005]

Here it is clear how a commercial constraint is considered a limitation for the implementation of a new feature, nevertheless the same constraint is still considered fundamental, and reworked with adjustments and revisions of expectations. In that way, the relationships between the commercial side of Solaris and the FLOSS effort in OpenSolaris, are discussed and negotiated, and also used as a comparison with the wider software industry panorama. The open development model, and in its differences with the proprietary software (in the case of Apple) or its differentiation among FLOSS projects (in the case of Linux), is discursively performed by developers. Generally speaking, this performance of the relationships among OpenSolaris and other entities construct the commercial side of FLOSS, mainly in its organizational peculiarities.

5. DISCUSSION

In this paper we have argued that in literature there is often a mistaken separation “FLOSS vs. commercial”. We have also argued that this point of view will need to be overcame, and that this can be done by looking at the practices of FLOSS development at large, practices that mixes different gradients of FLOSS and commercial activities everyday. From this point of view we need to remember that “commerce” can mean at least two different things: first, it is a “*social intercourse*” involving the “*interchange of ideas, opinions, or sentiments*”; second, it is “*the exchange or buying and selling of commodities on a large scale involving transportation from place to place*” (Merriam Webster on line). From this point of view, we can describe commercial practices as double – sided:

on one side, they are the interchange of ideas and opinions, on the other side they can involve the commodification of products and their trade.

It is interesting to see that especially the first definition of "commerce" seems to adapt fairly well to the practices of FLOSS, probably even more than what we have called the "*proprietary and closed source software*". As much as the same FLOSS development model is based on a continuous circulations of ideas, solutions and even people (Weber, 2004) we can clearly assign to it the nature of commercial.

More important than what we think is, however, what FLOSS developers (and FLOSS stakeholders in general) think and do. In the case of GRASS, for example, the commercial character of FLOSS is clearly linked with the provisions of the GNU GPL V.2 (in particular the term 1) that allow the distribution of copies of the software for a fee. We have described the emergence, around the term 1 of negotiations and controversies on the definition of GRASS as commercial software. At the same time we observed that the commercial nature of GRASS is used as an element to enrol more developers in the community. In addition, it emerges that the use of commercial to characterize only proprietary software is strongly opposed by GRASS developers.

In the case of OpenSolaris, the commercial side of FLOSS is performed through two different arenas of action: first, the construction of an OpenSolaris-based distribution that is provided by Sun to its customers; second, through the translation of the pre-existing proprietary software development, influenced by commercial aims, into the community software development practices. In both those arenas the intersection, between the commercial nature of Sun and the FLOSS development strategy, is able to raise controversies and negotiations on the role of the corporation, taking the commercial side for granted.

6. CONCLUSION

We would like to conclude this paper with a reflection prompted by a quotation from Philip Dick's work. In his introductory essay to the collection of short stories *I Hope I Shall Arrive Soon & Other Stories*, Dick discusses two recurring themes in his work: "what is the real man?" and "what is reality?". It is the second of these themes that is of particular interest to us, insofar the contribution of our paper can be seen as a contribution to answering the question: "what is the reality of FLOSS?". The phrase we would like to quote from Dick's work is the following: "*The basic tool for the manipulation of reality is the manipulation of words. If you can control the meaning of words, you can* *control the people who must use the words.* " (Dick et al., 1985).

In the essay Dick argues that the reality is not simply something that is out-there, ready to be discovered or used at will. By contrast, reality appears to be something that is "manufactured" (though often by an improper use of media) along with the way we act or think in and about it. This consideration of reality as something constructed by our actions, rather than something that is simply given, even lead him to consider that we should speak not just about reality, in singular terms, but about realities in plural terms. What Dick seems to argue in the phrase we have quoted is that the manipulation of word is not just a neutral process that does not influence the external reality. Words are not just neutral elements that simply represent real things that compose the reality out-there. Rather, words are the bricks we use to "manufacture" and sustain a specific definition of reality. Therefore, the manipulation and the control of a certain set of discursive practices is also related to the ability to manufacture a certain reality: discourse and power are deeply interleaved, if not the same thing. This is a lesson – the identity between words and worlds, in contrast to the view that sees words as mere representation - that the French philosopher and sociologist Micheal Foucualt has taught us in his famous essay *The Order of Things* (1966).

This brief digression in Dick's view about reality helps us in better framing what we have tried to discuss in this paper: the relationship between the word commercial and FLOSS reality in the light of the fact that the academic world has often assumed them as an antonym. The view we propose on this issue is very close to the Actor-Network Theory (ANT) approach in Science and Technology Studies (see in particular Law, 2004) in which there is the proposal that we should speak about realities in plural terms. Moreover, these realities are often seen in ANT as the result of construction processes that involves an array of human and non-human elements (Latour, 1999), deeply interlinked in heterogeneous networks of power (Latour, 1987). Dick's argument on realities (often referred to as *ontological politics* (Mol, 1999) in ANT) is more suitable for the conclusion of this paper insofar it help us to emphasize the role of words as part of this realities construction process.

Indeed, as Dick seems to argue, the process of manufacturing and constructing realities is grounded on the dynamics revolving around the meaning of words. What we have described in this paper is the process thorough which the use of the word "commercial" in relation with FLOSS and "proprietary software" is subject to this same type of dynamics. It appears that often (this is clear in the case of GRASS) FLOSS developers want to free the word "commercial" from the control of specific discursive practices and discursive uses. This because this word is meant, by FLOSS developers, to be used to build a specific definition of reality in which the antonym of FLOSS is not commercial but proprietary software. By contrast, it appears that it is the proprietary world that somehow seeks to control the use of the world commercial either in the text of software licenses or in generic discussions. In other terms, from FLOSS developers point of view, it seems that the proprietary world wants to manufacture a reality in which FLOSS is portrayed as the antonym of commercial. By contrast FLOSS developers want to manufacture a reality in which FLOSS is seen as commercial software. In this "game" of opposition it is clear that who can control the meaning of the word "commercial" is also in the best position to manufacture the reality he/she prefers.

At this point there is an even more important issue that we need to emphasise: the role of academic discourse. It seems that scientific publications often uncritically assume FLOSS as an antonym of commercial software. Given the above considerations about the role of words in building realities, we should reflect on whether the academic discourse participates to a specific construction of reality, rather than being the manifestation of a supposedly *neutral* point of view. We have to ask ourselves which reality we contribute to manufacture by being academic and researchers in FLOSS. In our opinion, the academic discourse fully contributes to a definition of reality in which FLOSS is sharply opposed to the word commercial and in which proprietary software is often portrayed as synonym of commercial. In his brilliant essay Wheeler (2006) seems to argue that this is just a confusion and a mistake, because writer are "s*imply unable to understand what is happening in the software industry*". As academics we should ask ourselves some questions, in particular if it is true that we are unable to understand what is going on with FLOSS. Again, Philip Dick vision on realities can help us. Indeed we can argue that the way the word commercial is used by the academic discourse around FLOSS might not just be a simple misunderstanding, but rather it might reflect the control over the use of words itself. A legitimate question that we just want to ask is: "*is this use of the word commercial in the FLOSS academic discourse a misunderstanding, or is it the result of manipulation or is even something that actively contributes to the manipulation?*". Although we currently do not have an answer to this questions, we think that possible answers lies in empirical research that is informed by an ethno-methodological view of FLOSS stakeholder. We think that this is the better instrument that we have for understanding the phenomena in which we are interested in.

REFERENCES

Akrich, M. (1992). The De-Scription of Technical Objects. In *Shaping Technology, Building Society: Studies in Sociotechnical Change*, W. Bijker and J. Law (eds.). Cambridge, MA: MIT Press.

Bessen, G. (2005). Open Source Software: Free Provision of Complex Public Goods. Research on Innovation paper URL: <http://researchoninnovation.org/opensrc.pdf>. [retrieved June 10th, 2009]

Bitzer, J. (2004). Commercial versus open source software: the role of product heterogeneity in competition. In *Economic Systems*, 28(4), p. 369-381.

Bonaccorsi, A. and Rossi, C. (2002). Why Open Source software can succeed. In *Research Policy*, 32(7), p. 1243-1258, URL: <http://linkinghub.elsevier.com/retrieve/pii/S0048733303000519> [retrieved June 10th, 2009]

Callon, M. (1986). Some elements of a sociology of translation: domestication of the scallops and the fishermen of Saint Brieuc Bay. In *Action and Belief: a new Sociology of Knowledge?*, cur. da. John Law, p. 196 - 233. London: Routledge and Kegan.

Ciborra, C. (2002). *The Labyrinths of Information.* Oxford: Oxford University Press.

De Paoli, S. (2008). *Software, Copyright e Pratiche di Licensing.* Unpublished Doctoral Dissertation, University of Trento (Italy)

De Paoli, S., Teli M., D'Andrea V. (2008). Free and open source licenses in community life: Two empirical cases. In *First Monday.* URL: <http://firstmonday.org/htbin/cgiwrap/bin/ojs/index.php/fm/article/view/2064>. [retrieved June 10th, 2009]

De Paoli, S. (2009). *The Archaeology of the Internet: on using FLOSS archives in social research*. Unpublished Manuscript, National University of Ireland Maynooth.

Dick, P. K., Hurst M., and Williams P. (1985). *I hope I shall arrive soon*. Doubleday.

Free Software Foundation,. (2004). *The Free Software Definition*. <http://www.gnu.org/philosophy/free-sw.html>. [retrieved June 10th, 2009]

Free Software Foundation, (1991). *GNU/ General Public License License 2.0*, URL: <http://www.gnu.org/copyleft/gpl.html>. [retrieved June 10th, 2009]

Foucault, M. (1970). *The Order of Things.* London: Tavistock.

Garfinkel H (1967). *Studies in ethnomethodology* . Englewood Cliffs, NJ: Prentice-Hall.

Godfrey, M. W. and Tu, Q. (2000). Evolution in Open Source Software: a case study. In *IEEE Proceedings of International Conference on Software Maintenance*, 2000, p. 131-142, URL<http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=883030&isnumber=19102> [retrieved June 10th, 2009]

Goldman, R. and Gabriel R. P. (2005). *Innovations Happens Elsewhere. Open Source as Business Strategy*. San Francisco: Elsevier.

GRASS Development Team, (1999-2006a). *GRASS History*, accessed 22 May 2007, URL<http://grass.itc.it/devel/grasshist.html>. [retrieved June 10th, 2009]

Koch, S. and Schneider, G. (2002). Effort, co-operation and co-ordination in an open source software project: GNOME. In *Information Systems Journal*, 12(1), p. 27-42), URL<http://www3.interscience.wiley.com/cgi-bin/fulltext/118925737/HTMLSTART>. [retrieved June 10th, 2009]

Kogut, B., and Metiu, A. (2001). Open Source Software Development and Distributed Innovation. In *Oxford Review of Economic Policy,* 17(2), pp. 248-264.

Healy, K. and Schussman, A. (2003). The Ecology of Open-Source Software Development, URL:<http://opensource.mit.edu/papers/healyschussman.pdf>. [retrieved June 10th, 2009]

Heidegger, M. (1927). *Being and Time*, trans. Macquarrie J. & Robinson E. (London: SCM Press, 1962).

Hertel, G. Niedner, S. and Herrmann, S. (2003). Motivation of software developers in Open Source projects: an Internet-based survey of contributors to the Linux kernel. In Research Policy 32(7), p. 1159-1177, URL: <http://www.sciencedirect.com/science?_ob=ArticleURL&_udi=B6V77-48BV04S-2&_user=107385&_rdoc=1&_fmt=&_orig=search&_sort=d&view=c&_version=1&_urlVersion=0&_userid=107385&md5=6df05bba608bfed32355d9af357114e9>. [retrieved June 10th, 2009]

Lanzara, G.F. and Morner, M., (2005). Artifacts rule! How organizing happens in open source software projects. In *Actor-Network Theory and Organizing,* Czarniawska B. and Hernes T. (Eds.), Liber e Copenhagen Business School Press, pp. 67-90.

Latour, B. (1987). *Science in Action*. Cambridge, MA: Harvard University Press.

Latour, B. (1999). *Pandora's Hope.* Harvard University Press, London.

Latour B. (2005). *Reassembling the Social. An Introduction to Actor-Network-Theory.* New York: Oxford University Press.

Law, J. (2004). *After Method: Mess in Social Science Research.* London: Routledge.

Lerner, J. and Tirole, J. (2005). The Scope of Open Source Licensing. In *Journal of Law, Economics, and Organization,* 21(1), p. 20-56.

Lin, Y. (2005). The future of sociology of FLOSS. In *First Monday,* Special Issue #2: Open Source, October 2005, URL < http://firstmonday.org/issues/special10_10/lin/index.html>. [retrieved June 10th, 2009]

Mockus, A., Fielding, R. T., and Herbsleb, J. D. (2002). Two case studies of open source software development: Apache and Mozilla. *ACM Trans. Softw. Eng. Methodol.* 11, 3 (Jul. 2002), p. 309-346.

Mol, A. (2002). *The Body Multiple, Ontology in Medical Practice*, Duke University Press, Durham NC.

Perens, B. (2005). The Emerging Economic Paradigm of Open Source. In *First Monday* Special Issue 2.URL: <http://www.firstmonday.org/issues/special10_10/perens/index.html>. [retrieved June 10th, 2009]

Raymond, E.S. (1999a). The Magic Cauldron. URL: <http://www.catb.org/esr/writings/magic-cauldron/> [retrieved June 10th, 2009]

Raymond, E.S. (1999b). *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary.* Sebastopol, California: O'Reilly and Associates.

Teli, M. (2008). *Libertà e Pratiche. Il software libero e open source nel caso OpenSolaris*. Unpublished Doctoral Dissertaion, University of Trento (Italy)

Teli, M., Pisanu, F. and Hakken, D. (2007). The Internet as a Library-of-People: For a Cyberethnography of Online Groups [65 paragraphs]. *Forum Qualitative Sozialforschung / Forum: Qualitative Social Research*, 8(3), Art. 33, http://nbn-resolving.de/urn:nbn:de:0114-fqs0703338 [retrieved June 10th, 2009]

Von Hippel, E., and Von Krogh G., (2003a). Special Issue on Open Source Software Development. In *Research Policy,* 32(7), p. 1149-1157, URL: <http://linkinghub.elsevier.com/retrieve/pii/S0048733303000544>. [retrieved June 10th, 2009]

Von Hippel, E., and Von Krogh, G. (2003b). Open source software and the" private-collective" innovation model: Issues for organization science. In *Organization Science,* 14(2), p. 209-223.

Weber, S. (2004). *The Success of Open Source*. Cambridge MA. Harvard University Press.

Wheeler, D. A. (2006). Commercial” is not the opposite of Free-Libre/Open Source Software (FLOSS). December, URL <http://www.dwheeler.com/essays/commercial-floss.html>[retrieved March 23rd, 2009]

Winograd, T. and Flores, F. (1986). Understanding Computer and Cognition. Norwood, NJ: Ablex.