PsychNology Journal, 2007
Volume 5, Number 3, pp. 299 – 309

Market Relations, Non-Market Relations and Free Software

Andres Baravalle ♠♦ and Sarah Chambers ♣

♦ Dept. of ICT,
The Open University,
Milton Keynes (UK)

♣ Dept. of Computer Science,
The University of Sheffield,
Sheffield (UK)

ABSTRACT

Free Software is sometimes considered solely a technical option, but that is a quite limited point of view: we suggest, indeed, that Free Software is not merely a technical option, but it is, in fact a different working paradigm for the software development community and a different model for acquiring (and sharing) resources in the Information Society. This paper will discuss this working paradigm and analyse the market and non-market relations that are implied by it.

Keywords: *free software, open source, business models, hacker ethics, software engineering*

Paper Received 26/07/2008; received in revised form 28/12/2007; accepted 28/12/2007.

1. Introduction

Researchers from different backgrounds have been analysing market and non-market relations in different contexts: amongst Stone Age tribes (Sahlins, 1972), amongst the Trobriands in Melanesia (Malinowski, 1984) and in the modern industrial society (Lapavitsas, 2004), just to name a few examples. These researches, while different in many aspects, show how there has been and there still is a strong presence of non-market relations in different societies. This paper aims to show how a similar perspective can be seen in the Information Society context, discussing the role of the Free Software movement.

Cite as:
Baravalle, A., & Chambers, S. (2007). Market Relations, Non-Market Relations and Free Software. P*sychNology Journal, 5*(3), 299 - 309. Retrieved [month] [day], [year], from www.psychnology.org.

♠ Corresponding Author:
Andres Baravalle
The Open University, Walton Hall
Milton Keynes, United Kingdom, MK7 6AA
E-mail: a.baravalle@open.ac.uk

The Free Software movement advocates the development and use of software that is free to use, free to modify and free to redistribute, on both pragmatic (Stallman, 2002) and philosophical (Stallman, 2004) grounds. The Free Software movement shares many points of contact with the Open Source movement (Perence, 1999), and they are often commonly referred together, as FLOSS (Free/Libre and Open Source Software) (FLOSS project 2002), FOSS (Free and Open Source Software) (Bollinger, 2003) or F/OSS (MIT Open Source Group, 2002).

In this paper we are referring to Free Software, but the discussion applies equally to Open Source, as the two terms are, in most cases, interchangeable.

Free Software is not necessarily gratuitous; freedom to use refers to the freedom to run for any purpose, not to price. In some cases, the user may have to pay a price to obtain the software: for example, Richard Stallman was charging 150 dollars for the first version of Emacs (Stallman, 1988).

Similarly, some software gives the user the freedom to use and redistribute the software, but not to change it and thus it does not qualify as Free Software. This is the case, for example, of freeware and shareware (Werbach & Dreben, 2003), that are merely royalty-free.

Noticeable examples of Free Software include the operating system Linux, the web server Apache and the office suite OpenOffice.org.

Research on Free Software has already examined in depth the business models that drive its development (Bonaccorsi & Rossi, 2003; Haruvy, Prasad, & Sethi, 2003; Hertel, Niedner, & Herrmann, 2003; Lakhani & Wolf, 2003; Lerner & Tirole, 2001; Lerner & Tirole, 2002). Some organizations base their business on selling services (e.g. IBM, Novell and Sun). For example, often companies sell Linux distributions, that are based on the customization and integration of existing Free Software. Other companies, such as MySQL AB, do business by selling proprietary versions of their Free Software, that can be integrated with the proprietary software of their customers, while their Free Software version can be integrated with Free Software only. Finally, other organizations are supported by users that buy Free Software (e.g. Lsongs or PhoneGaim), that can be legally obtained without paying; in this case buying the software is mainly a voluntary way of supporting the developers, or a way to avoiding the hassle of having to obtain the software in a different way which usually involves searching for the software with the use of web search engines, or compiling the software from the source code.

In this paper we aim to analyse Free Software from a different point of view, focusing on *the market and non-market relations inside the community of users and developers of Free Software*.

Some authors, as Raymond (Raymond, 1999), see Free Software mainly from a technical, pragmatic point of view, as a way to obtain better software. Others (Victor, 2003; Moglen, 1999; Benkler, 2002) consider Free Software as the seeds of a different kind of Information Society, that goes beyond the classical model of market economy. We insert our work in this latter group, trying to extend the discussion with our contribution.

The following sections will discuss labour, capital and product and their respective role in the production paradigm of Free Software.

2. Labour and Capital

Pure market values, as money-oriented rewards, are often insufficient to motivate many workers (Herzberg, Mausner, & Snyderman, 1959): once material needs have been met, an increase in the availability of material rewards is no longer sufficient. Instead, fulfilment of higher level needs within the hierarchy of needs (Maslow & Lowry, 1998) is a required to motivate the work force.

This means that esteem and self actualisation needs have to be considered, and that the worker cannot be motivated only by monetary means. The Free Software community is pervaded by what has been defined as *hacker ethics* (Levy, 1984), and this needs to be considered for motivating the developer. It is important to note that in the context of this paper, as in the context of the Free Software community, hacker is neutral term referring to skilled software developers.

The values of passion, freedom, work ethic, money ethic, network ethic, caring and creativity, as described by Himanen (2001), pervade the hacker ethics and the Free Software developers. Freedom is valued as itself, more than any alternatives values, as the hacker ethics *refuses to bow to pure market or convenience*:

“But just because we are competing with proprietary software on issues of technical merit doesn't mean we think people should choose the program for source control based on technical qualities alone. That would mean assigning zero value to freedom itself. If you value freedom, you will resist the temptation to use a program that takes away your freedom, whatever technical advantages it may have” (Biancuzzi, 2004).

Hacker ethics resembles the chieftainship ethics, as described by Malinowsky (1984). The chieftain, as the hacker, is linked to his tribe (his community) by duties and privileges – he can ask for tributes, in terms of work or resources – but his prestige and his ability of claiming work and resources, depend on his contribution to the community. Hackers with a good reputation will be able to involve people in their projects (and in many projects, a huge amount of developers is required), and people will be honoured to be chosen. In exchange, the hackers need to contribute and to demonstrate their commitment to the community. The influence of a hacker that does not contribute to the community will be effectively reduced. This is similar in other communities, as in the scientific community: the scientists with highest status are the ones who have directly and indirectly contributed the most to their fields, not those who possess the most knowledge.

On the other hand, Free Software developers (very often) have commercial interests in software development. This is true for almost all the important Free Software developers, at least at some point of their career. Linus Torvalds, the creator of Linux, has been working at Transmeta (a chip manufacturer) for more than 6 years before moving to the OSDL lab (a non profit organisation). Guido Van Rossum, the author of Python, has been working for several different companies in the past years. Miguel de Icaza, the main developer of Gnome, founded his own consultancy company. At the same time, Free Software developers work inside a community and most of the product of their labour is available to the community. The result is *a dual allegiance of developers,* both to the community and to their companies.

According to Stallman (Laird, n.d.) in the Free Software community there is a moral priority of the community on the business. Nevertheless, many companies (including corporations like IBM or Novell) support the development of Free Software. The interest of corporations is motivated not in the name of ethics, but of market values exclusively, and it could not be differently (Bakan, 2004).

It is clear that the motivations for the support of Free Software are completely different for the labour force and for the capital: non-market values from one side (the hacker ethics), market values from the other. At the present time, each one needs the other and collaboration is the only viable solution to fulfil the self interest of each one: companies can hire developers, and they are free to work in the context of a community that endorses their values.

3. Product

Software and its documentation are distributed according to different types of licensing conditions. The Free Software community itself is using different licenses (the so-called Free Software licenses), according to specific requirements, but they all provide four freedoms (to execute, to study, to redistribute, to improve) to the users, as defined by the Free Software Foundation (Stallman, 2004).

Three main licenses are used in the Free Software community: the GPL (GNU Public License) (Free Software Foundation, 1991), the LGPL (Lesser General Public License) (Free Software Foundation, 1999) and the BSD (Berkeley Software Distribution) license (Open Source Initiative, 2005). The GPL requires that the rights that a user received with a GPL software must be granted by any program derived or linked to it, which means from a practical point of view, that developers including GPL software in their projects must release their work *under the same conditions*. The BSD license allows inclusion of the Free Software in proprietary software. The LGPL is used mainly by libraries (shared components), and allows to use it in any type of software, but requires that any modifications to the software are released under the same conditions.

Free Software licenses do not restrict the user from using Free Software in a non-free operating system (for example users can run Firefox, or OpenOffice.org on a non-free operating system as Microsoft Vista). Nevertheless, Free Software licenses as the GPL or the LGPL can put restrictions on derived works (for example, you cannot alter the Linux kernel and release your changes as proprietary software).

In July 2005, we performed an analysis of the software at SourceForge (sourceforge.net), the widest existing repository of Free Software (including at the time more than 65,000 Free Software projects) to study the diffusion of the different licenses. At that time, we found that nearly 69% of the software included was released under the GPL license, 11% under the LGPL and 7% under the BSD license. A previous analysis by Wheeler (Wheeler, 2005) showed similar results: in 2003, 71% of SourceForge Free Software projects (45,000) were using the GPL, 10% the LGPL and 7% the BSD license. It needs to be noted that the software can be released (and often is) under more than one license and these data refers to software that included the GPL as possible license. Furthermore, in Wheeler's earlier analysis (Wheeler, 2002) of Red Hat Linux 7.1 (Red Hat was and still is the most popular Linux vendor) he found that nearly 50% of its code was released under the GPL only.

This analysis shows how around 70% of the developed Free Software is released under the GPL, which does not allow to integrate the software into proprietary programs, and that an additional 7% can be used in proprietary software but any improvements need to be released as Free Software. The implication is that Free Software and proprietary technology currently cannot be easily mixed. Users can use Free Software and/or proprietary applications, but in the vast majority of cases Free Software applications cannot have proprietary components and vice versa.

The implication is that the Free Software community is creating a niche in which market and non-market relations can develop only between members – companies that do not develop Free Software cannot use Free Software component, and often are not able to market their products to the users of Free Software.

In many cases software developers say that the software is developed “just for fun” (Torvalds & Diamond, 2001). As a consequence, Free Software is characterised by a *precarious usefulness*, as addressed by the GPL itself, in clause 11:

> “there is no warranty for the program, to the extent permitted by applicable law. [...] parties provide the program 'as is' without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose.”
>
> (Free Software Foundation, 1991).

Similar clauses are inserted both in the LGPL and in the BSD license.

This strongly supports considering Free Software as permeated with non-marked relations. For this aspect (but not for others, as will be discussed in the next section) a *similitude with the gift* is possible. A gift is given and accepted within a loose evidence of usefulness, and Free Software is distributed “as is” - and may or may not be useful for the user.

Nevertheless, the usefulness of a product is individual, and often it can be maximised, in the users of Free Software, by its freedom.

The usefulness of software, for a user, is strictly linked to the satisfaction of some of the needs of a user. A computer game will be useful if it is providing leisure, and a business application will be useful if it is suitable for the business processes of its user. Free Software is likely to have a very different degree of usefulness for a hacker and for a user that does not know how to use and configure a GNU/Linux system. On the other hand, its precarious usefulness itself opens the way to market relations: companies can provide services, supporting or customising software that by itself might not be suitable to the end user.

4. A different Organization Paradigm

Free software is often considered an element of a gift economy, where goods and services are not traded, but freely given and taken. Gift economies can usually develop in the presence of trust or abundance. When based on trust, individuals share the certainty of reciprocity and will freely give away their goods, or provide service, knowing that they would be able to receive back from the community what needed, at due time. Gift economies based on abundance develop when there is no scarcity of goods and the social status is determined not by what the individuals' controls but by what is given away.

Eric Raymond is one of the supporters of the view of Free Software as a good exchanged using the rules of a gift economy (Raymond, 1999). This perspective is based on the idea that software is a non-scarce good that can be easily shared, because the cost of replication for software tends to zero and thus "the only available measure of competitive success is reputation among one's peers".

However, it can be disputed from a number of different angles that Free Software is not an element of a not gift economy based on trust nor is it based on abundance.

First of all, it is hard to consider Free Software as an element of a gift economy, because a gift requires the property of an inalienable item to be exchanged (Gregory, 1982), while with Free Software no property is exchanged. Proprietary software can be considered as elements of a gift economy, not Free Software.

Moreover, a gift economy is always based on the presumption of reciprocity inside the community, sometimes described as a "delayed market exchange", and there is always a penalty for failing. The Free Software production model is, indeed, based on reciprocity between users and developers, but of a different kind: it is *voluntary*, *delayed* and *tending to immaterial for the developer*.

It is voluntary because it is not obligatory to provide any sort of regard to the author of a Free Software that is used. It is possible and recommended to do so, however it is not mandatory, and even if you disrespect Free Software, you can use it (as long as you respect the license conditions - otherwise you may lose the right to use the software). The example of the SCO case (Shankland, 2003) is meaningful: SCO has been attacking the GPL for several years, as part of their controversy against Linux (Galli 2003) but nevertheless this does not imply that they cannot use GPL software (they are still using and distributing GPL software in their products).

The reciprocity is delayed, because the regard is not usually given for the access to the software, but after the use of the software itself, if the users appreciate it. The developer of software may not get any form of immediate regard for producing Free Software. It may happen, in some cases, when Free Software is produced within the horizon of a business project; for example Free Software may be commissioned by a customer, or may be developed as part of the commitment of an organisation to a Free Software project, or may be due for realising a service that is to generate revenues. In other situations, the regard will be delayed in time, if and when the use the software will grow. To point out the difference, the regard for the production of proprietary software is generally monetary and strictly linked to the access to the software itself. While software companies may publish demo versions or development versions of software, for the full, working software a fee is required in most cases. On the other hand, while in a gift economy the regard is linear, in the Free Software model the regard is exponential, thanks to the free circulation of the software.

Moreover, the reciprocity is tending to immaterial, because once a user acquires a copy of a Free Software, it can be given to any number of other users, and its price quickly tends towards zero. In a market economy, the cost of a product is usually decomposed as the cost of the product itself and the cost of transfer. Buying in a shop a t-shirt has a cost of transfer, that is the cost of shipping the t-shirt from its manufacturing location to the shop, and the cost of manufacturing. In a gift economy, a product has a value (based on cost of production and cost of transfer), but it is given for free, on the basis that the gift will be reciprocated. In the case of Free Software, any user that acquires (for a price or for free) a copy of the software can redistribute it, for a price or for free. This implies that in most cases the value will quickly tend towards zero, and that a immaterial regard may be the only regard that the developer is receiving from most users.

When we receive something with a non-market transaction, we do not pay not the product nor the transfer in exchange for the good. This is the case of Free Software, but as we have seen we are facing non-market relations that do not appear in the context of a gift economy.

By contrast with the non-market relations that happen in a gift economy, *we can define the Free Software community as a socioeconomic system in which the access to goods do not require an agreement upon a quid pro quo, reciprocation or regard, but obedience to a defined set of rules which define the possible use of the good.*

We can as well foresee how production numbers that tends towards to infinite and a common access to non scarce products that can be easily shared would overcome the dilemma posed by the capitalist model of production on how to distribute a scarce production. Thus, the Free Software model implies a different approach to the scarcity of resources and to the diffusion of innovation.

We have already seen how companies have to accept the hacker ethics to be part of the Free Software community, not very different from Venetian merchants accepted in the Feudal economy. Free Software is not a collateral experiment, or a small community, and if we accept that infrastructure is depending from the production structure, we may foresee a change in the organisational structures that are involved in the development of software, and we may have the seeds for different production models for the Information Society.

5. References

Bakan, J. (2004). *The corporation: The Pathological Pursuit of Profit and Power*. London: Constable & Robinson Ltd.

Benkler, Y. (2002). *Coase's Penguin, or Linux and the Nature of the Firm*. Retrieved September 1, 2005, from ttp://www.yale.edu/yalelj/112/BenklerWEB.pdf

Biancuzzi, F. 2004. *Freedom, Innovation, and Convenience: The RMS Interview.* Retrieved September 1, 2005, from http://www.linuxdevcenter.com/pub/a/linux/2004/12/22/rms_interview.html

Bollinger, T. (2003). *Use of Free and Open-Source Software (FOSS) in the U.S.* Retrieved September 1, 2005, from www.egovos.org/rawmedia_repository/588347ad_c97c_48b9_a63d_821cb0e8422d?/document.pdf

Bonaccorsi, A., & Rossi, C. (2003). Why Open Source Software can succeed. In *Research Policy*, *32*, 1243-1258.

FLOSS Project (2002). *Free/Libre and Open Source Software: Survey and Study*. Retrieved September 1, 2005, from http://flossproject.org/report/

Free Software Foundation (1999). *GNU Lesser General Public License*. Retrieved 1 September, 2005, from http://www.gnu.org/licenses/gpl.html

Free Software Foundation (1991). *GNU General Public License*. Retrieved September 1, 2005 from http://www.gnu.org/licenses/gpl.html

Galli, P. (2003). *SCO Group Launches Broadside Against GPL*. Retrieved December 28, 2007, from http://www.eweek.com/article2/0,1895,1404984,00.asp

Gregory, C. 1982. *Gifts and Commodities*. London: Academic Press.

Haruvy, E., Prasad, A., & Sethi, S.P. (2003). Harvesting altruism in Open Source Software development. In *Journal of Optimising Theory and Applications*, *118*(2), 381-416.

Hertel, G., Niedner, S., & 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*, 1159-1177.

Herzberg, F., Mausner, B., & Snyderman, B. B. (1959). *The motivation to work*. London: J.Wiley.

Himanen, P. (2001). *The Hacker Ethic and the Spirit of the Information*. London: Random House, Incorporated.

Laird, C. (n.d.). *How free is open? An interview with Richard M. Stallman*. Retrieved September 1, 2005, from http://www.developer.com/open/article.php/603961

Lapavitsas, C. (2004). Commodities and gifts: Why commodities represent more than market relations. In *Science & Society*, *68*(1), 33-56.

Lakhani, K., & Wolf, R. (2005). *Why hackers do what they do: Understanding motivation and effort in Free/Open Source Software Projects*. Retrieved September 1, 2005, from http://ssrn.com/abstract=443040

Lerner, J., & Tirole, J. (2001). The Open Source Movement: Key research questions. In *European Economic Review*, *45*, 819-826..

Lerner, J., & Tirole, J. (2002). Some simple economics of Open Source. In *The Journal of Industrial Economics*, *50*(2), 197-234.

Levy, S. (1984). *Hackers: Heroes of the Computer Revolution*. USA: Bantam books.

Malinowski, B. (1984). *Argonauts of the Western Pacific*. Illinois, USA: Waveland Press.

Maslow, A., & Lowry, R. (1998). *Toward a Psychology of Being*. New York: Wiley & Sons.

MIT Open Source Group (2002). *Why Open Source?* Retrieved September 1, 2005, from http://opensource.mit.edu/what_is_os.html

Moglen, E. (1999). *Anarchism triumphant: Free software and the death of copyright*. Retrieved September 1, 2005, from http://firstmonday.org/issues/issue4_8/moglen/index.html

Open Source Initiative (2005). *The BSD License*. Retrieved 1 September, 2005, from http://www.opensource.org/licenses/bsd-license.php

Perens, B. (1999). The Open Source Definition. In C. Di Bona, S. Ockman, & M. Stone (Eds.), *Open Sources: Voices from the Open Source Revolution*, (pp. 171-189). O'Reilly Media.

Raymond, E. S. (1999). *Homesteading the Noosphere*. Retrieved September 1, 2005, from http://www.firstmonday.org/issues/issue3_10/raymond/

Sahlins, M. (1972). *Stone age economics*. Chicago : Aldine-Atherton.

Shankland, S. (2003) . *SCO attacks open-source foundation*. Retrieved September 1, 2005, from http://news.com.com/SCO+attacks+open-source+foundation/2100-7344_3-5098610.html

Stallman, R. (2002). Copyleft: Pragmatic Idealism. In J. Guy (Ed.), *Free Software, Free Society: Selected Essays of Richard M. Stallman* (pp. 91-95). Boston, MA: Free Software Foundation.

Stallman, R. (1988). *The GNU Project.* Retrieved on December 28, 2007 from http://www.gnu.org/gnu/thegnuproject.html

Stallman, R. (2004). *The Free Software Definition*. Retrieved September 1, 2005, from http://www.fsf.org/philosophy/free-sw.html

Torvalds, L., & Diamond, D. (2001). *Just For Fun: The Story of an Accidental Revolutio*. USA: Texere Publishing.

Victor, R. (2003). *Free Software and Market Relations*. Retrieved September 1, 2005, from http://www.oekonux.org/texts/marketrelations.html

Werbach, J. L., & Dreben, R. N. (2003). The Accidental Licensor: Advanced Issues in Software Licensing. In *ACCA Docket*, *21*(2), 54-71.

Wheeler, D. A. (2002). *More Than a Gigabuck: Estimating GNU/Linux's Size*. Retrieved September 1, 2005, from http://www.dwheeler.com/sloc/redhat71-v1/redhat71sloc.html

Wheeler, D. A. (2005). *Make Your Open Source Software GPL-Compatible. Or Else*. Retrieved September 1, 2005, from http://www.dwheeler.com/essays/gpl-compatible.html