Ubuntuism, Commodification, and the Software Dialectic

Mike Chege

September 23, 2008

Abstract

"Free as in speech, but not free as in beer," is the refrain made famous by Richard Stallman, the standard-bearer of the free software movement. However, many free software advocates seem to be of the opinion that the purity of free software is somehow tainted by any preoccupation with money or profit. Inevitably, this has implications for the economic sustainability of free software, for without a source of income, how can free software hope to survive? The challenge of finding a way to ensure economic sustainability without sacrificing the ideals of free software is what we have termed the "software dialectic."

While the literature on the economics of free software is already quite substantial, in this essay we approach the subject by considering first principles, such as the difference between wealth and money, how the market chooses what to produce, and what the laws of thermodynamics have to do with economics. Finally, even though the ideas expressed in this essay apply to free software in general, here we shall be focusing on that particular incarnation of free software known as the GNU/Linux system.

1 Introduction

The old saying that there is no such thing as a free lunch is so trivial and obvious that it seems hardly worth mentioning. Yet, to paraphrase DeLong (2004), free software advocates often treat the topic of money like the Victorians treated sex: everyone knew that sex must exist to produce little Victorians, but any talk about sex was considered verboten.

DeLong's analogy might seem a little exaggerated, but there is no denying that money has long been a touchy subject in free software circles, and even though attitudes have mellowed in recent years, you still run the risk of wearing out your welcome among free software advocates by using words like money and profit too freely.

On the other hand, contrary to the image often painted of him as a head-in-the-clouds, anti-commercial ideologue, Richard Stallman, the standard-bearer of the free software movement, seems to take a rather pragmatic view. According to him, free software is not opposed to the making of money. As he writes:

> Many people believe that the spirit of the GNU project is that you should not charge money for distributing copies of the software, or that you should charge as little as possible - just enough to cover the cost.
>
> Actually, we encourage people who distribute free software to charge as much as they wish or can. (Gay, 2002, p. 65)

According to Stallman, the money earned from selling the software can then be used to support the growth of free software:

> Distributing free software is an opportunity to raise funds for development. Don't waste it!
>
> In order to contribute funds, you need to have some extra. If you charge too low a fee, you won't have anything to spare to support development. (Gay, p. 65)

Perhaps one of the reasons why free software advocates exhibit such reticence about the subject of money is that free software is based on a philosophy of inclusion, cooperation, sharing, and openness, while the market is synonymous with self-interest, secretiveness, competition, and the exclusion of those who do not, or cannot, pay. The challenge of reconciling these two contradictory opposites is what constitutes the "software dialectic."[1]

Since this is a discussion that raises fundamental economic questions, it will also be a discussion about first principles, namely the difference between wealth and money, how the market chooses what to produce, and what the laws of thermodynamics have to do with economics.

Finally, even though the ideas expressed in this essay apply to free and open source software projects in general, here we will be focusing on the GNU/Linux system.[2] It should also be noted that even though the terms free software and open source software are often used interchangeably in the literature, the truth is that the terms open source and free software refer to separate movements with different goals. Since in this essay we are concerned

[1] Here we use the term "dialectic" not in the Socratic sense, but in the sense that the German philosopher Georg Hegel and his most famous disciple, Karl Marx, used it. According to Hegel and Marx, the dialectical process, comprising the stages of thesis, antithesis, and synthesis, was a general law governing not just society but nature itself. Though their ideas have long since been abandoned by most scholars, the Hegelian/Marxian dialectical framework may still serve a useful purpose. In this essay, we use the dialectical framework to describe the attempt to reconcile two contradictory positions (thesis and antithesis) in order to arrive at an outcome that accommodates both positions (the synthesis).

[2] The GNU in GNU/Linux is a recursive acronym for "GNU's not Unix." The GNU project was the first major project of the free software movement. Initiated in 1984 by Richard Stallman, the goal of the GNU project was to create a complete, free as in freedom, operating system, consisting of a kernel, assemblers, compilers, text editors, mailers, and other programs. By 1990 the GNU system was almost complete but lacked one major component, the kernel. In 1991, Linus Torvalds, an undergraduate student at the University of Helsinki, developed a Unix-compatible kernel and called it Linux. Combining Linux with the GNU system resulted in a complete free operating system. The name GNU/Linux is therefore meant to express the composition of the system as a combination of the GNU system with Linux as the kernel while also ensuring that the ideal of freedom, which led to the creation of the GNU system in the first place, is not forgotten. An account of the history of the free software movement, the GNU project, and Linux can be found in Gay (2002, pp. 17-28).

with the economic as well as the social and ethical aspects of free and open source software, we shall be making a distinction between free and open source software.

2 The Midas Fallacy

Most of us are familiar with the Greek legend of King Midas. To many it is a cautionary tale about the dangers of avarice. For economists, however, the story of King Midas may hold another lesson. This is that wealth and gold (or money) are not the same thing. After all, here was the world's richest man in danger of dying from hunger and thirst no different from a destitute beggar on the streets. The king was in danger of dying from hunger and thirst because everything he touched turned to gold, which meant that food and drink was also rendered inedible. Clearly, when Midas asked Dionysus for the power to turn everything he touched into gold, he was not interested in gold as an end in itself, but rather in the things that gold could buy.

The tendency to confuse money with wealth is what we are referring to as the Midas Fallacy. Wealth may be defined as goods and services that satisfy material human wants and needs, while money simply represents a convenient if imperfect way to measure and exchange wealth. Or to put it in the far more euphonious prose of the 18th century Scottish philosopher David Hume:

> Money is not, properly speaking, one of the subjects of commerce; but only the instrument which men have agreed upon to facilitate the exchange of one commodity for another. It is none of the wheels of trade: it is the oil which renders the motion of the wheels more smooth and easy. (Robinson & Eatwell, 1973, p. 6)

Now, while anybody in King Midas's position might probably be more concerned about how to convert money (gold) into wealth (food and drink for a start!), in this essay we are more concerned with how to convert wealth (namely free software) into money, which brings us to the subject of commodification.

3 Commodification, Property Rights, and the Exclusion Principle[3]

Economists classify goods into two categories: private and public. The market mechanism is well suited to the provision of private goods. The market mechanism is based on exchange and exchange can occur only where there is an exclusive right of ownership to the property which is to be exchanged. In fact, the market system may be viewed as a system for the exchange of property rights.

To put it another way, the market can function only in a situation where the "exclusion principle" applies, i.e., where A's consumption is made contingent on A's paying the price, while B, who does not pay, is excluded. Exchange cannot occur without property rights, and property rights require exclusion. The market therefore tends to cater to those particular consumer demands that are amenable to the exclusion principle. This bias means that in situations where exclusion would be impossible or prohibitively expensive, no goods or services will be provided by the market mechanism, even if the provision of such goods or services is socially desirable, and a situation known as "market failure" thus occurs.

A classic example of market failure is national defence. Since it is impossible to provide the benefits of national defence to those of the country's inhabitants who are prepared to pay while excluding the benefits from those who are not prepared to pay, each individual will find it in his or her interest

[3] Much of the credit for the ideas expressed in this section must go to Fred Hirsch (1978) whose essay, "The New Commodity Fetishism," inspired our use of the commodification concept. Elucidation of the commodification concept was also greatly aided by Richard and Peggy Musgrave's text (1989, pp. 42-44).

to consume without paying, or to "free ride." Without the ability to exclude free riders, the incentive to provide national defense through the market thus disappears and it is for this reason that certain services, such as national defence, must be provided by the government.[4]

So what is commodification? Commodification may be described as the process by which the market establishes property rights and excluding devices to facilitate the production and exchange of wealth. In the software industry, excluding devices may include legal devices such as software licenses, and physical devices such as copy restriction and product activation codes.[5] For our purposes, we shall also extend the commodification concept to include the methods by which firms seek to establish property rights over knowledge so as to gain an advantage over potential competitors. Such methods may include secrecy, patents, and copyrights, as well as attractive pay and working conditions where knowledge is embodied in (i.e., carried around in the heads of) employees.

4 Ubuntuism, or the Freedom to Help Your Neighbour

If the antithesis of exclusion is inclusion, then the antithesis of commodification is ubuntuism. The term ubuntuism is a neologism coined from the

[4] Government of course does not provide such goods and services for free. Since the power of coercion is an essential feature of government, it can effectively counteract the free-rider problem by requiring all who receive the benefits of a public good or service to pay appropriate taxes for it.

[5] The physical devices mentioned are an attempt to turn software into a rival good. A rival good is a good whose consumption by one person diminishes its availability for others. For instance, when I eat an apple it is no longer available for anyone else. Or, if I sit on a chair, that chair is no longer available to anyone else to sit on until I relinquish it. In contrast, software on a CD or DVD can, like the biblical fish and seven loaves that fed the multitude, be duplicated indefinitely or installed on an infinite number of machines, so that my use of the software does not diminish its availability to others. Hence, in an attempt to make consumption of software rivalrous, software publishers may introduce such techniques as copy restriction of media and product activation codes.

southern African word Ubuntu which is related to a Zulu concept, "umuntu ngumuntu ngabantu," which means that a person is only a person through their relationship to others.[6] This may also be expressed as “I exist because the community exists.” Because of its emphasis on “others” and “community,” Ubuntu may be described as a form of communitarianism. Communitarianism is a system of social organization that stresses the significance of mutual reciprocity, charity, fairness, openness, and a commitment to the common good. We shall be using the term ubuntuism to refer to the communitarian philosophy in general.

Incidentally, lest we come away with the impression that ubuntuism is some kind of romantic ideal, it should be noted that most historians and anthropologists are in agreement that communitarian social organization has in fact been the predominant form of social organization through almost all of human history and may date back millions of years to early ancestors such as the Australopithecines (Stavrianos, 1990, pp. 17-23).[7] This is also consistent with the insights of sociobiology which show that in the struggle for survival, a strategy based on cooperation and mutual support rather than selfishness and opportunism would be the optimal strategy.[8]

Since this is a discussion about free software, one might ask, what is the link between ubuntuism and free software? The link between ubuntuism and free software lies in the ethic of sharing, mutual support, openness, and social solidarity which characterizes both ubuntuism and the free software movement. In the free software movement this ethic is encapsulated in Richard Stallman’s famous Four Freedoms which are the guiding principles of the

[6] The term Ubuntu has gained prominence in recent years in part due to its popularisation by major public figures such as Nelson Mandela, Desmond Tutu, and even Bill Clinton (Coughlan, 2006, September 28). For a description of the Ubuntu way of life in a traditional African setting, see Xulu (2007, November). Within the free and open source software communities, the term Ubuntu is mostly associated with the popular Ubuntu Linux distribution.

[7] Though Stavrianos refers to the system as “kinship society,” it is essentially communitarianism by another name.

[8] See Heylighen & Campbell (1995) for a discussion on the evolution of cooperation.

movement. These Four Freedoms are as follows:[9]

Freedom 1: The freedom to run the program, for any purpose.

Freedom 2: The freedom to study the source code, and change it to make the program do what you wish.

Freedom 3: The freedom to redistribute copies so you can help your neighbor.

Freedom 4: The freedom to improve the program, and release your improvements to the public, so that the whole community benefits.

To quote Stallman (McGregor, 2008) on the importance of the Four Freedoms: "[The Four Freedoms] give you control over your own computing, and make it possible to be part of a community where people help each other.... These values of freedom and social solidarity are the basic ideals of the free software movement, ever since its founding in 1983. These ideals were my motive for launching the development [of] the GNU operating system. They are the reason our community exists."

The ethical underpinnings of free software are what distinguishes the free software movement from its offshoot, the open source movement. For the open source movement, the issue of whether software should be open source is a practical question, not an ethical one. As its proponents frankly admit, the concept of open source software was conceived with the explicit goal of making free software more appealing to the business community by citing the practical benefits of the open source development methodology.[10] Proponents of free software, on the other hand, tend to see the ethical and social values as being more important. As a result, although nearly all open source software is free software,[11] the free software movement and the open source movement are essentially separate movements with different views and goals.

[9] Based on Stallman (Gay, p. 43).

[10] See for instance Raymond (1999).

[11] That is, software that is licensed under the GNU General Public License or GPL, the copyright license that gives legal expression to the Four Freedoms. We would also add software licensed under GPL-compatible licenses, such as the Modified BSD license, the Perl license, and certain Python licenses.

Still, regardless of the differences between the two movements, the ideas expressed in this essay are applicable to both movements because both movements have to grapple with the question of how to maintain economic sustainability while largely forfeiting the opportunity to earn money the old-fashioned way - by charging license fees.[12]

5 A Model of Economic Entropy

Let us now turn to the concept of entropy which is another concept that is of central importance to our approach. We shall start by providing a simplified (but hopefully, not simplistic) explanation of the entropy concept.[13]

Energy, it has been said, is the currency of the universe. However, the flow of energy in the universe is subject to a set of physical laws known as the laws of thermodynamics. Here we are interested in the second law of thermodynamics, also known as the law of entropy. The second law of thermodynamics states that the "entropy" of a "closed" system always increases. This means that such a system will spontaneously "run down" as all available energy is used up and eventually there will be no potential for further useful work. A system which no longer has the capacity to do useful work because all the energy available to it has been used up is said to be in a state of

[12] Free and open source software does not completely shut the door on the possibility of earning money through licensing fees. Companies can make money from licensing fees by resorting to a dual licensing strategy. This involves offering their software under two different licensing schemes, a proprietary licensing scheme and a free or open source licensing scheme. However, as Bruce Perens (2005) explains, dual licensing is better suited to software libraries and generally isn't usable for applications, which therefore limits the number of situations in which dual licensing may be applicable. In the final analysis, however, the mixing of proprietary and free and open source licenses does seem to run counter to the concept of free and open source software.

The same objection would also apply to any strategy based on the bundling of proprietary software with free and open source software as a way of promoting a free and open source software product.

[13] Here we are mostly concerned with thermodynamic entropy. The entropy concept has also been extended to non-thermodynamic contexts. For a good discussion on the different contexts in which the entropy concept has been applied, see Joslyn (1990, August).

maximum entropy.

A useful analogy to describe the second law is an hour glass.[14] An hour glass can be considered a closed system in that no sand enters the glass and none leaves. But although the quantity of sand in the hour glass is constant, the bottom chamber is filling up and the top chamber is becoming empty. The sand in the bottom chamber may be seen as a measure of the amount of entropy in the system. Sand in the top chamber is capable of doing work by falling, like water at the top of a waterfall, while sand in the bottom chamber has spent its capacity to do work. The second law therefore states that whenever work is done the amount of usable energy in the system declines, which, in the case of a closed system, means that the system will eventually run down. Furthermore, though an hourglass can be flipped over, the second law of thermodynamics states that in any physical process entropy never decreases. To put it another way, the original inputs may never be recovered from the outputs of the process and any attempt to recover the inputs would itself entail the expenditure of more energy and materials which would themselves be irrecoverable, and so on, ad infinitum. The irreversibility of physical processes is what leads to entropy's designation as "time's arrow."

Coming to "open" systems, such systems, unlike closed systems, receive a continuous inflow of material and energy from their environment so that they do not run down. In the terminology of the physicist Erwin Schrödinger (1945), open systems receive "negative entropy" from the environment. Biological systems such as human beings are good examples of open systems.

Because every process of production is, at bottom, a transformation of energy and matter, it should come as no surprise that a number of economists have found the laws of thermodynamics to be concepts with considerable relevance for economics. In fact, interest in the laws of thermodynamics has

[14] This analogy appears in an excerpt from Daly & Cobb (1994) on the DieOff website (Dieoff). The analogy is attributed to the economist Nicholas Georgescu-Roegen.

led to the rise of an approach known as thermoeconomics.[15] Methods within thermoeconomics range from from those that seek to develop highly technical analytical models of the economy based on the laws of thermodynamics, to those that view thermodynamic concepts as analogies and metaphors. While the analogies-and-metaphors approach may not allow us to make exact and deductive scientific statements about economic systems, it still has merit as a heuristic, with the capacity to allow us to see economic phenomena in a new light and hence stimulate research in new and potentially fruitful directions. In this essay we will be taking the analogies-and-metaphors approach.

So, how do we incorporate the entropy law into out model? If we acknowledge that any process of economic production is dependent on close interaction with the environment from which it extracts materials and energy, then it follows that any process of economic production that remains isolated from its surroundings, in the sense that it does not receive any inputs from its surroundings, will eventually succumb to "economic entropy." In other words, it will eventually become incapable of generating any more outputs just as a thermodynamic system that has used up all its energy is incapable of doing any more work.

Now, we should make it clear that when we talk of a closed economic system we are not talking of a closed system in the physics sense, that is, a completely self-contained system that does not exchange any matter or energy with its environment, very much like the hour glass in our analogy. Instead, we are talking of a quasi-closed system (see Figure 1 below) which, in addition to exchanging outputs with its environment, may receive inputs from its environment, but these "exogenous" inputs (represented by the arrow with a dashed line in Figure 1) are so limited relative to the material and

[15] Or, alternatively, bioeconomics. Nicholas Georgescu-Roegen is considered to be one of the pioneers of thermoeconomics as a distinct approach within economics with his monumental work *The Entropy Law and the Economic Process*, published in 1971. A brief yet informative introduction to thermoeconomics is provided by Baumgärtner (2004). Other works worth consulting are Gowdy & Mesner (1998) and Corning (2002).

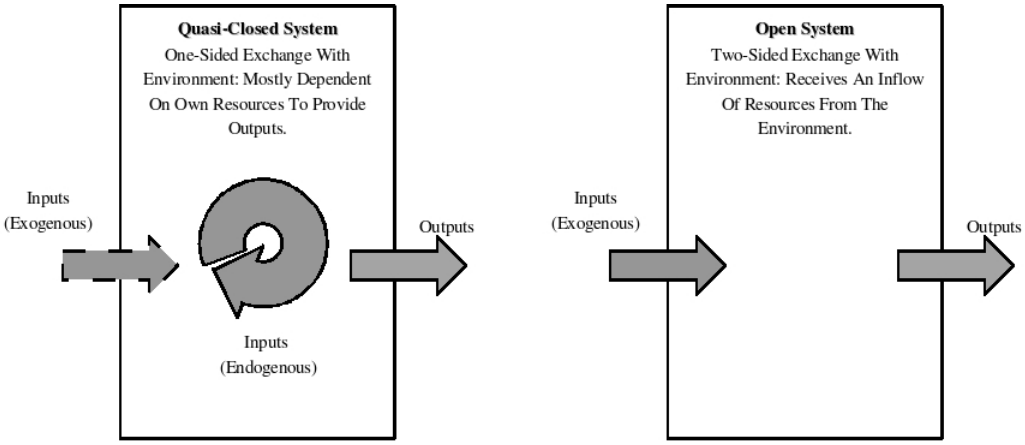

Quasi-Closed System
One-Sided Exchange With Environment: Mostly Dependent On Own Resources To Provide Outputs.
Inputs (Exogenous)
Outputs
Inputs (Endogenous)
Open System
Two-Sided Exchange With Environment: Receives An Inflow Of Resources From The Environment.
Inputs (Exogenous)
Outputs

Figure 1: Closed and Open Systems of Economic Production.

energy requirements of the system that the system has to rely mainly on its own internal resources or "endogenous" inputs, which are presumed to be finite, which in turn means that the system faces the risk of running down sooner or later.

The next question is, what constitutes inputs in our model? According to the production function, $q=f(x_1,x_2 \ldots x_n)$, which is the standard way of representing the production process in economic models, inputs typically comprise labour, capital (both financial and physical), and raw materials.[16] One could therefore say, that as long as there is a steady and sufficient supply of these inputs in any production process, then economic entropy can be avoided.

[16] Where q is the quantity of output and $x_1,x_2 \ldots x_n$ are the quantities of the inputs used. f is simply the mathematical notation for "is a function off" or "depends upon." Of course, the production function concept is not without its critics (see for example Baumgärtner [2004, p. 117] in the context of thermodynamics, and Mishra [2007] and Cohen & Harcourt [2003] in the context of conventional economic analysis), but for present purposes such criticisms need not concern us.

6 Closed System and Open System Distros

Having outlined a concept of economic entropy, let us now apply the concept to the GNU/Linux system which is the focus of this essay.[17]

We generally encounter the GNU/Linux system in the form of a "distribution." So it would be in order to start by explaining what a distribution is. A distribution, or distro, to use the popular term, is an operating system comprising the Linux kernel and an assortment of applications in binary form but also available in source code form. Now, in principle, users could obtain the Linux kernel via the Internet and they could then obtain source code for the applications in the same way, and then compile and integrate the programs to assemble a working GNU/Linux system. However, this process is both very demanding and time-consuming. To avoid the hassle of "rolling" their own distributions, therefore, users often choose to obtain complete, ready-made systems from "distributors," the companies or individuals that undertake the task of creating such systems.

GNU/Linux distributions come in all shapes and sizes. They range from small, single-CD, single-purpose distros like SystemRescueCd (whose main task, as its name suggests, is to "rescue" systems after a crash) to massive, everything-but-the-kitchen-sink, multiple-CD distros like Debian. Distros can also be distinguished along the lines of software package formats, that is, whether they are rpm-based distros, deb-based distros, or tgz-based distros. But distros can also be distinguished along the lines of whether they are closed system distros or open system distros.

In the previous section we described a closed system as a system which may receive inputs from its environment but these exogenous inputs are so limited relative to the material and energy requirements of the system

[17] Though not all the software in the GNU/Linux system is licensed under the GPL, free software forms a significant part of the GNU/Linux system and this includes some of the most essential components such as the Linux kernel itself, without which there would be no system. For more on the rationale behind the name GNU/Linux, refer to note 1.

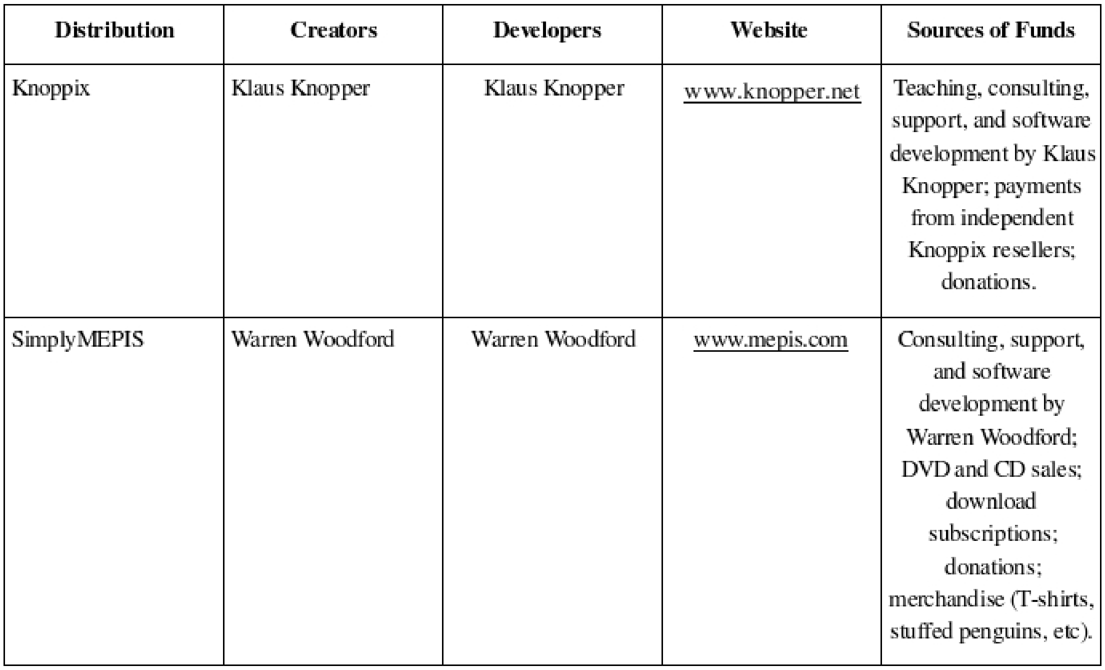
that the system has to rely mainly on its own endogenous inputs which are however assumed to be finite, which means that the system faces a high risk of running down.

Since building and maintaining a distro is an intrinsically labour intensive activity, an admittedly unsophisticated but, in our view, effective way to measure the level of "openness" of a distro is to look at the number of developers involved in the project. By this standard, a typical closed system distro will have a very small group of developers, even just one or two developers, often the creators of the project, working on it. In addition, alongside the low developer headcount, we are also likely to find that the majority of the resources available to the project are being provided by the developers themselves. In other words, there is greater reliance on internal sources of funds. All in all, even assuming that they were possessed of prodigious programming abilities, the amount of time, effort, and resources that a small group of developers can devote to a distribution is necessarily finite. This means that closed system distributions must contend with the ever-present prospect of entropic decline.

Coming to open system distros, such distros by contrast will be able to draw on the environment for developers as well as other resources. An open system distro, therefore, is likely to have considerably more developers working on it in addition to having better external sources of funds.

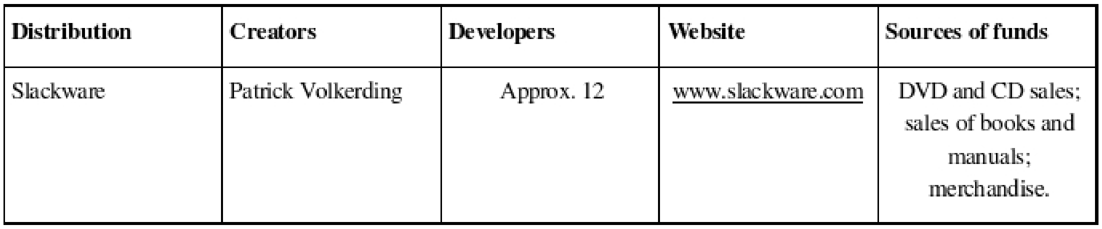
Using the number of developers and the sources of funds as our main yardsticks, in the tables below we present a selection of the most popular distros divided into closed system and open system distros.[18]

[18] Information on how many developers are involved in a distribution is generally hard to come by. We have only included those distributions for which we have fairly reliable information. Information on the Slackware distribution is from an interview with Patrick Volkerding, the creator of Slackware, on The Linux Link Tech Show (TLLTS, 2006). The information on Knoppix comes from a search on "Knoppix" and "Klaus Knopper" on the free and open source software news site, Distrowatch (Distrowatch-a), and the Wikipedia entries on Knoppix (Wikipedia-a) and Klaus Knopper (Wikipedia-b). Information on the Mepis distribution comes from an interview with Warren Woodford, the creator of Mepis, on the Linux Link Tech Show (TLLTS, 2007). Details on Mandriva are from the Mandriva



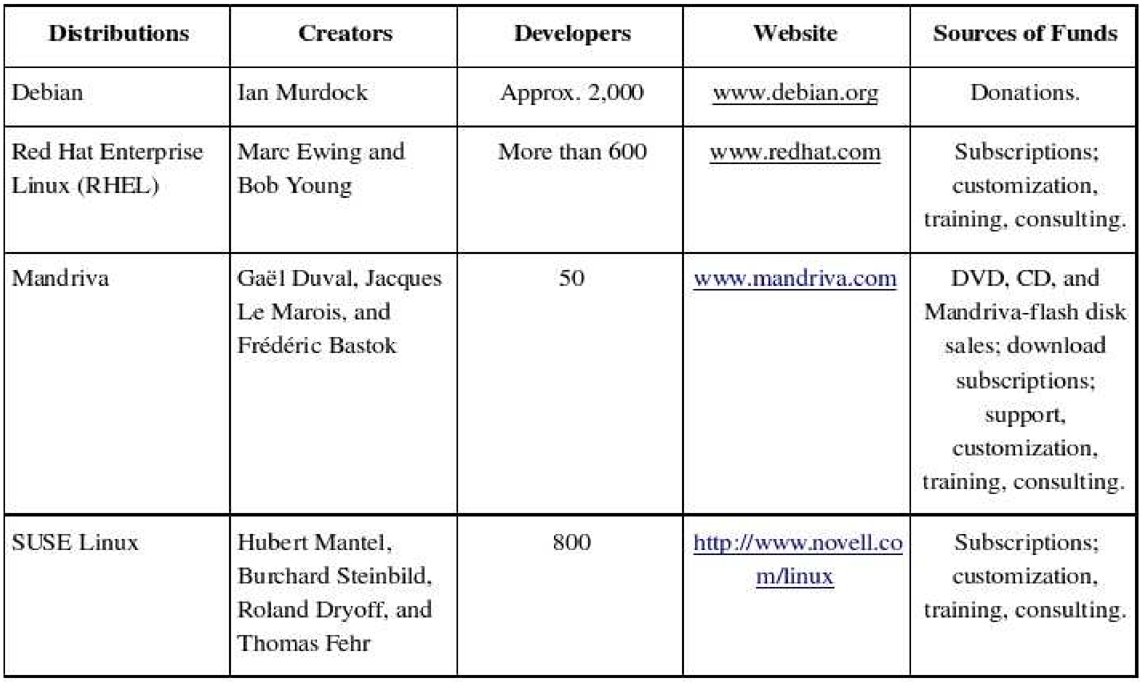
Distribution	Creators	Developers	Website	Sources of Funds
Knoppix	Klaus Knopper	Klaus Knopper	www.knopper.net	Teaching, consulting, support, and software development by Klaus Knopper; payments from independent Knoppix resellers; donations.
SimplyMEPIS	Warren Woodford	Warren Woodford	www.mepis.com	Consulting, support, and software development by Warren Woodford; DVD and CD sales; download subscriptions; donations; merchandise (T-shirts, stuffed penguins, etc).

Table 1: Closed System Distros.



Distribution	Creators	Developers	Website	Sources of funds
Slackware	Patrick Volkerding	Approx. 12	www.slackware.com	DVD and CD sales; sales of books and manuals; merchandise.

Table 2: Borderline Open System Distros.



Distributions	Creators	Developers	Website	Sources of Funds
Debian	Ian Murdock	Approx. 2,000	www.debian.org	Donations.
Red Hat Enterprise Linux (RHEL)	Marc Ewing and Bob Young	More than 600	www.redhat.com	Subscriptions; customization, training, consulting.
Mandriva	Gaël Duval, Jacques Le Marois, and Frédéric Bastok	50	www.mandriva.com	DVD, CD, and Mandriva-flash disk sales; download subscriptions; support, customization, training, consulting.
SUSE Linux	Hubert Mantel, Burchard Steinbild, Roland Dryoff, and Thomas Fehr	800	http://www.novell.com/linux	Subscriptions; customization, training, consulting.

Table 3: Open System Distros.

In many cases closed system distros have become synonymous with their creators. Thus, it may be difficult to imagine Knoppix without Klaus Knopper or Mepis without Warren Woodford. While this may be part of the charm of a closed system distro, it can also be a serious shortcoming because whether there is a new release or not may be heavily dependent on the creator's situation. Thus, for example, for some time Klaus Knopper has been working on ADRIANE, a desktop system designed for blind computer users (Knopper's wife, Adriane, is blind).[19] As a result, Knopper has had less time to work on the "regular" Knoppix which has affected the frequency of new releases.[20] As for Mepis, since 2005 Warren Woodford's work on the distribution has been slowed down by a back problem.[21] More recently, when new concerns were raised about the future of Mepis given the prolonged lack of news on the distribution, Woodford disclosed that work on Mepis had suffered because he "finally had to reenter the workforce as a consultant in order to pay the bills. I can [earn] more in two weeks of consulting, then in a year with [Mepis]" (Vaughan-Nichols, 2007). Woodford's candid admission also lends more support to our characterization of closed system distros as lacking adequate external sources of funds.

Meanwhile, we have referred to Slackware as a borderline open system distro (Table 2) rather than a fully open system distro. This is because even

corporate information web page (Mandriva-a), while details on the Debian and Red Hat distributions are from Debian.net and Red Hat Vice President and CTO Brian Stevens's keynote address at the 2007 Red Hat Summit (Red Hat, 2007b) respectively. Details on the SUSE Linux distribution are from Stoveland (2008, p. 19). Information on sources of funds was gathered from the distributions' websites.

[19] While Knopper obviously named the ADRIANE desktop system after his wife, ADRIANE is also an acronym for Audio Desktop Reference Implementation and Networking Environment, as explained on the Knopper.net website (Knopper.net).

[20] There was whole year between releases from March 2007 (version 5.2) to March 2008 (version 5.3), as opposed to the usual 3-5 month release cycle.

[21] In an interview on the Linux Link Tech Show (TLLTS, 2007), Woodford revealed that he had been suffering from a herniated disc in his back for around three years. This had affected his productivity but unfortunately he did not have the money to pay for the expensive surgery needed to fix the problem.

though Slackware makes enough money to support its creator "Pat" Volkerding (TLLTS, 2006), and has a "development team" with about a dozen developers (most likely volunteers), the evidence suggests that Volkerding still does most of the development work himself, and it might be just as difficult to imagine Slackware without Volkerding as it might be to imagine Knoppix without Knopper or Mepis without Woodford. Ultimately, because of their dependence on a handful of individuals or just one individual, the chances are high that closed system distros and borderline open system distros may cease to exist if these individuals, for whatever reason, are unable to continue working on them.[22]

Fully open system distros, on the other hand, tend to develop a separate identity and existence from their creators. For instance, even though the Debian distribution still bears the name of its creator Ian Murdock (the "ian" in Debian),[23] the project has long since acquired a life of its own. Similarly, Red Hat is no longer tied to its founders Mark Ewing and Bob Young,[24] and Mandriva even fired one of its founders (Mandriva, 2006, March)!

The last few years have seen a massive proliferation of GNU/Linux distros (there are estimated to be over 300 "active" GNU/Linux distributions),[25] most of which we would probably classify as closed system distros, and it is common to hear people decrying the proliferation of distros as a wasteful

[22] This was the fate that befell Libranet GNU/Linux, a popular commercial distribution with a team of four developers. Designed as an easy-to-install Debian-based distribution back in the days when installing and configuring Debian was still something of an ordeal, Libranet ran into problems following the demise of its creator, Jon Danzig, in June 2005. Tal, Danzig's son, attempted to keep the distribution going but eventually abandoned the task (Danzig, 2005, November 25).

[23] The "Deb" in Debian comes from the first name of Murdock's girlfriend (now wife) Debra.

[24] According to Yahoo!Finances (Yahoo!Finances), Mark Ewing and Bob Young are now neither major shareholders nor key executives in the company.

[25] For a list of known distributions refer to Distrowatch (Distrowatch-b) and LWN.net (LWN.net). An "active" distribution is a distribution that is still being maintained and updated.

duplication of effort[26] and a cause of unnecessary confusion and frustration to users.

Such proliferation may be described as a "widening" process. This widening process in itself needn't be a bad thing. After all, the freedom to tinker with code is supposed to be one of the main freedoms, and strengths, of free software. But it could also be argued that if GNU/Linux is to solidify its position as a viable alternative to proprietary or non-free software, then "deepening," which implies a greater focus on qualitative improvements, as opposed to mere quantitative increase, is also needed. Such deepening may involve adding important, new functions to software, or it may simply involve making existing functions more robust and efficient.

Now, while many, if not most, of the developers working on closed system distros are undoubtedly very talented and innovative people, it appears to us that deepening, especially the kind that involves making fundamental improvements to the software, is more likely to be undertaken by open system distros because such distros, almost by definition, are in a better position to carry out the skill- and resource-intensive activity of R&D. As an example, in the fiscal year ending February 29, 2008, Red Hat spent $97.4 million on R&D (Red Hat, 2008, p.18), a figure well beyond the capability of any closed system distro. Meanwhile, Novell, which owns the SUSE Linux distribution, assigns large groups of developers to work on complex software packages such as the OpenOffice office suite, the KDE desktop system, and the Linux kernel (Stoveland 2008, p. 69). Once again, this is beyond the capacity of any closed system distro.

Let us now try to pull together the threads of our argument by looking at two major open-system GNU/Linux distros, each pursuing a different strategy in the attempt to overcome economic entropy.

[26] See, for instance, Vaughan-Nichols (2005, September 29).

7 Red Hat: A Case Study in the Commodification of Free

Red Hat is the proverbial garage startup. In 1994, Marc Ewing, who had recently graduated from Carnegie Mellon University, created his own version of Linux working out of his spare bedroom. He named his version Red Hat Linux. Marc's release of his initial beta of Red Hat, the "Halloween" edition, earned him instant fame in Linux newsgroups. Shortly afterwards, Bob Young, owner of the ACC Corporation, a small catalog business that sold Linux and UNIX products, called Ewing up because he wanted to add Red Hat to his catalog of Linux products. A few months later, Young and Ewing agreed to merge Marc's Linux business with Ewing's ACC corporation, and Red Hat Software Ltd. was born.[27]

Of course, Red Hat was not the only distribution available at the time. There were already other distributions such as SLS, Yggdrasil, and Slackware. But Red Hat's selling point was that it developed a software program to aid in the complicated package installation and upgrade process. That program was the Red Hat Package Manager or RPM in short. Thanks to RPM, coupled with efforts to make the system easier to install and configure, Red Hat's popularity soared.

From the outset, however, it must have been clear that selling software as a product was not going to be an easy way to build a sustainable business. This was because while the GNU General Public License or GPL, the copyright license that gives legal expression to the Four Freedoms and under which the Linux kernel and much of the software in the GNU/Linux system is licensed, has no requirements about how much you can charge for distributing a copy of free software,[28] it also allows, even encourages, users to copy and distribute the software. People could thus redistribute copies of Red Hat

[27] Red Hat Software Ltd. was changed to Red Hat, Inc. in June 1999.

[28] In Stallman's words (Gay, p. 72), "you can charge nothing, a penny, a dollar, or a billion dollars."

without a penny coming to Red Hat.[29] Furthermore, because competitors had access to the same freely available source code as Red Hat, there were limits to what Red Hat could charge for its product. As Bob Young (Martin, 2007) put it (brackets appear in original quote): “You couldn’t make any money selling [the Linux] operating system...because all this stuff was free, and if you started to charge money for it, someone else would come in and price it lower.”[30]

Given the difficulty of surviving on software product sales alone, Red Hat had to search for a new business model. Despite the growing popularity of the GNU/Linux system, one of the biggest obstacles to GNU/Linux adoption in the corporate environment was a lack of support services. Seeing this as an opportunity, Red Hat begun to look into providing post-sales support to corporate clients. This marked the switch to what may be described as the “sell religion, not bibles” strategy with Red Hat moving up the software value chain from a business based largely on software sales (bibles) to one that was based largely on the sale of ancillary services related to its software (religion).

As shown in Figure 2, we have divided the software value chain into two sections: software packaging and software services. Software packaging refers to the traditional model of selling software as a product. Since we are talking about GNU/Linux distributions, software packaging would involve the aggregation, integration, and optimization of the Linux kernel and the numerous additional files that together form the GNU/Linux operating system, to create a distribution which may be accompanied by a manual and probably some basic post-sales support.

[29] In 2001 it was reported that of the 15 million to 20 million copies of the Red Hat distribution that were in the marketplace, only about 1.5 million copies had actually been purchased from Red Hat (Reason, 2001). This seems to belie Bob Young’s “Evian Water” argument (Young, 1999). According to this argument, for the same reason that are many people who would rather buy expensive bottled water when tap water is much cheaper, there were many people who would rather buy “Official” Red Hat Linux in a box for $50 rather than download it for free or buy unofficial CD copies for as little as $2.

[30] Here, Young seems to contradict his earlier Evian Water argument.

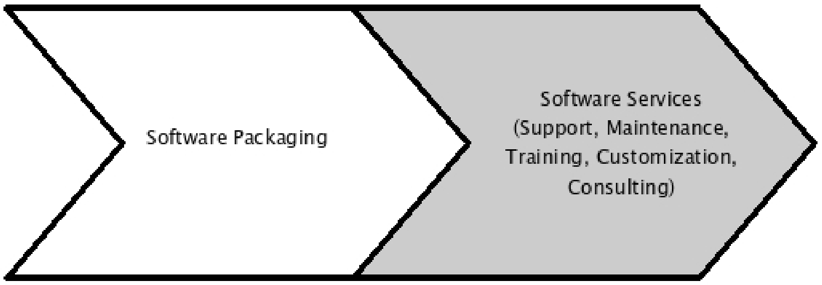

Software Packaging
Software Services (Support, Maintenance, Training, Customization, Consulting)

Figure 2: The Software Value Chain (Adapted from Berlecon Research [2002]).

Software services, on the other hand, refer to the provision of support, maintenance, training, integration, customization, and consulting services. Since GNU/Linux distributors have already built up technical competence from packaging and optimising the GNU/Linux system, the presumption is that they can capitalise on this know-how to provide support, consulting, and other services.

In 2003, as part of its new business strategy, Red Hat discontinued its retail edition of Red Hat and introduced Red Hat Enterprise Linux. Red Hat Enterprise Linux was to be sold on a subscription basis. A subscription entitled the user to a copy or copies of Red Hat Enterprise Linux, support services including installation and other technical assistance, and access to patches and updates through the Red Hat Network service.

So, to put Red Hat's experience within the framework of the software dialectic, the exclusion principle did not work well with software sales because free and open source licenses in general circumscribe the opportunities for earning money through license fees, the source code is freely available to competitors, and people are allowed to share and redistribute the software virtually as a right. With the services approach on the other hand, only those who buy a subscription are eligible to receive support from Red Hat, and even though adaptations of Red Hat Enterprise Linux such as CentOS can be downloaded for free off the Internet, access to support and maintenance services as well as updates to official Red Hat packages via the Red Hat Network are only available to subscribers.

The evidence suggests that this approach has paid off handsomely for Red Hat. In the fiscal year 2007-2008, Red Hat achieved revenues of $523 million and a net income (profit) of $76 million (Red Hat, 2008, p. 38). Total cash, cash equivalents, and investments as of February 29, 2008 were $1.3 billion, and at the time of writing, the company had a stock market capitalization hovering around $4 billion.

At the same time, Red Hat has always made an effort to remain on good terms with the free software community by making all of its source code available to the community and by supporting various free software projects.[31] Red Hat is also a founding member of the Open Invention Network which is a company that acts as a "patent pool" whose members have agreed not to assert their patents against GNU/Linux or GNU/Linux-related applications.[32]

Though Red Hat's behaviour may ultimately be guided by the less-noble goal of turning a profit and keeping its shareholders happy, it has attempted to do so while remaining scrupulously faithful to the letter, if not the spirit,[33] of the GPL.

[31] A list of projects that Red Hat supports can be found on the Red Hat "Open source development list" web page (Red Hat-a).

[32] Other founding members of the Open Invention Network include IBM, Novell, Philips, and Sony. More information can be found at the Open Invention Network website at http://www.openinventionnetwork.com/.

[33] It may be noted, for instance, that Red Hat refers to itself as an open source company and not a free software company. See for example the Red Hat "Investor Relations" web page (Red Hat-b). Also, as Bob Young has put it (Young, 1999), "...we are not ideological about licenses. We are comfortable with any license that provides us with control over the software we are using...."

8 Debian: The Apotheosis of Ubuntuism

On August 16th, 1993, Ian Murdock, an undergraduate student at Purdue University, announced the "Debian Linux Release" (Murdock 1993, August 16). The development of Debian came as a result of Murdock's disenchantment with GNU/Linux distributions at the time. According to Murdock, though many distributions had started out as fairly good systems, as time passed, attention to maintaining the distribution became a secondary concern (Murdock, 1994). Murdock gave the example of the Softlanding Linux System (SLS) which he described as possibly the most popular distribution at the time. Unfortunately, according to Murdock, SLS was also "quite possibly the most bug-ridden and badly maintained Linux distribution available." This meant that GNU/Linux users were being subjected to an inferior product and their bad experiences were bound to undermine the prospects of GNU/Linux. To Murdock, the time had come "to concentrate on the future of Linux rather than on the destructive goal of enriching oneself at the expense of the entire Linux community and its future." The primary purpose of the Debian project therefore was to create a distribution that would "live up to the Linux name" by being carefully and conscientiously assembled, maintained and supported.

Another important aspect of Murdock's plan was that unlike other distributions "which are developed by individuals, small, closed groups, or commercial vendors", Debian was to be developed cooperatively by many individuals through the Internet, in the same spirit as the Linux kernel and other free software. This "open process", Murdock believed, would ensure that the system was of the highest quality and that it reflected the needs of the user community rather than the needs and wants of the constructor.

Finally, for users who were not in a position to download the distribution from the Internet, Debian was to be made available on physical media at little more than cost, and any profits earned would be used to support the further development of free software.

It was an ambitious plan, but thanks to the hard work and enthusiasm of Ian Murdock and the others who succeeded him in the role of Debian Project Leader, Debian grew from a single PC under a student's desk at Purdue University, to paraphrase Bdale Garbee (2007),[34] to become a large, worldwide community of developers[35] and users. Currently, the project boasts roughly 2,000 developers in over 40 countries collaborating via the Internet, and even though no one gets paid to work on Debian, anecdotal evidence suggests that there is no shortage of applicants wishing to become developers.

Debian is also by far the largest GNU/Linux distribution. Based on source lines of code (SLOC) analysis, the estimated size of the latest release, Debian 4.0,[36] amounts to close to 283 million source lines of code (Amor, et al., 2007). Using the Constructive Cost Model (COCOMO) to estimate the effort and cost that would be involved in building a system the size of Debian 4.0 from scratch, we get an effort equivalent to 73,000 person years and a cost of around US$ 10 billion (Amor, et al., 2007). The Debian project therefore represents a monumental effort of wealth creation which is barely reflected in the economic statistics, a perfect example of the Midas Fallacy.[37]

But size is not the only noteworthy thing about Debian. Thanks to its reputation for stability, which owes much to its stringent quality assurance procedures,[38] many major organizations around the world now rely on

[34] Bdale Garbee was the Debian project leader between April 2002 and April 2003. While his comment was not meant to be taken literally, it is not far from the truth.

[35] The Debian project actually refers to its developers as "maintainers." However, for the sake of consistency, we shall continue to use the term developers.

[36] By the time of writing, Debian 5.0, which had been scheduled for release in September 2008, was still not ready for release.

[37] Actually, the figure for source lines of code includes both the work of the original or upstream authors, who in most cases are not Debian developers, and the work of Debian developers, which includes Debian-related scripts and patches necessary to adapt packages to Debian or to fix bugs, as well as Debian package management tools and installation software (Amor, et al., 2007, p. 5). It would therefore be more accurate to say that the US$ 10 billion figure represents the free software effort as a whole, or a good part of it at any rate, rather than just the Debian project's effort.

[38] As a now-famous post by a MrNemesis on Slashdot.com put it (Krafft, 2005, p. 103): "Look, this is Debian. They don't release things until you have to fire rockets at the thing

Debian. Examples include the Municipal Council of the City of Munich, the MIT Computer Science and Artificial Intelligence Lab, and the Genome Research Cluster at the Sanger Institute, which is the single largest contributor to the public Human Genome Project. Debian also powers the servers that run Second Life, the popular online virtual world, and more recently it has been reported that Debian also powers the Max Planck Institute's 32.8 teraflops supercomputer which is Germany's fourth largest supercomputer making it the 11th largest in Europe and the 34th largest in the world (Debian Times 2008, June 11). But perhaps most impressive of all is the claim by Bdale Garbee (2007) that there's a 30 percent chance that anyone making a cellphone call anywhere in the world will be relying on Debian in some way. This is because Hewlett Packard (HP), which is a major hardware vendor for telecommunications service providers, installs Debian on much of the hardware that it ships.

Debian, like free and open source software in general, appears to defy our preconceptions about how the economy should work. For here is a large and intricate software system which is being developed by a loosely-coordinated, globally-dispersed community of volunteers, it is being given away for free, and yet it is so highly regarded that major organizations are willing to trust it with some of their most critical operations. One could perhaps say that Debian is proof that the community model can work just as well as the commercial model. But questions about the sustainability of the community model still remain and we will have more to say about the subject shortly.

9 Free Software is Not a Perpetuum Mobile

The market mechanism is well suited to the production of goods where the benefits derived from a good flow exclusively to the person who paid for the good. Commodification is the name we have given to the process of

to stop it from working."

establishing property rights and excluding devices over goods and knowledge so that only those who pay for a good are entitled to its benefits while competing producers are denied the use of knowledge that would improve their competitive position relative to the firm possessing such knowledge. Due to its basis in exclusion, this process seems inimical to the spirit of free software which is the spirit of inclusion, cooperation, and sharing which we have termed ubuntuism. So the question is, how does one reconcile these two contradictory opposites, commodification and ubuntuism, so as to make free software economically viable while preserving the ideals of free software? This is the software dialectic.

As the example of Red Hat shows, it is possible to achieve some kind of synthesis between commodification and ubuntuism. To recap, given the difficulty of applying the exclusion principle to GNU/Linux as a product, Red Hat decided to move up the value chain to areas, namely software services, where the exclusion principle could be more successfully applied. At the same time Red Hat has continued to strive to remain on good terms with the free software community by contributing all its code back to the community in compliance with the GPL, and also by supporting various free software projects.

In contrast to the Red Hat model is the Debian model which appears to bypass the software dialectic entirely by relying on donations and volunteer developers. But is this a sustainable model?

There is already a substantial literature on why people may chose to work on free software without remuneration.[39] In a nutshell, people may be motivated by a desire to "scratch an itch" (i.e., to fix a bug or see some new function or feature added to the software), the enjoyment and intellectual stimulation of the activity, a sense of belonging and contributing to a cause,

[39] The canonical references are probably Raymond (1999) and Lerner & Tirole (2000). Other relevant works include Lakhani & Wolf (2005) and Rullani (2005). For a review about studies on motivations for contributing to free and open source projects see Rossi (2004).

a desire to improve one's programming skills, striving for recognition among peers, as a way to "signal" one's skills to prospective employers, or a combination of the above. If one assumes that Debian developers are no different from other free software developers, then one can probably say that as long as there are people willing to work on free software there will be people willing to work on Debian.

This last sentence betrays the fact that we have not been able to find any studies that shed light on the motivations of Debian developers specifically. However, there are other studies on volunteer participation in Debian which provide some insights which may be useful in assessing Debian's future prospects. For example, one of the main findings in Robles et al. (2005) is that there is a general stability in the volunteer effort in Debian. For a start, Debian developers tend to commit to the project for long periods of time: at least half are expected to remain with the project for seven and a half years, which is probably longer than in many software companies. Second, when developers leave the project most of their work tends to be taken over by other developers so that continuity is ensured. Robles et al. also note that there has been a trend towards the formation of developer teams to maintain packages. These developer or "maintainer" teams help to reduce the reliance on individual developers and minimize the impact that the departure of any single developer will have on the project.

Another pertinent study is the study by Michlmayr (2004). One of the observations that Michlmayr makes in his study is that over the years the admission process into the Debian project has become more elaborate and time-consuming. While some would argue that a more onerous admission process only serves to discourage new developers from joining the project, Michlmayr takes the view that the increased complexity of the admission process may not only act as a way of screening out volunteers who lack good technical skills, but it may also act as a way of screening out volunteers who are not likely to show a high level of commitment to the project. Though

Michlmayr offers some suggestions as to how this hypothesis might be tested, to our knowledge no such study has been undertaken so far.[40]

Besides stability in the volunteer effort, Debian has also been enjoying increased corporate and institutional interest.[41] This bodes well for the project because besides boosting Debian's profile, there is a good chance that corporations and institutions, whether out of a desire to "give something back to the community" or just out of sheer self-interest, may choose to provide support to the project, whether by making donations, hiring Debian developers, or contributing developers and code to the project, thus improving the project's long-term prospects.

On the whole, the continued success of the Debian project seems to support the view that with a well-defined vision that can galvanize people's interest and involvement, good leadership,[42] and the right organizational structures, the community model is just as viable as the commercial model, and just as capable of producing high quality software.

Alongside our discussion on the software dialectic we also had a discussion which used the entropy metaphor and the idea of closed systems and open systems to emphasize the fact that free software is not a *perpetuum mobile* or perpetual motion machine.

Since ancient times people have been trying to devise perpetual motion machines that would give us something for nothing. But so far the laws of thermodynamics have managed to frustrate all such efforts. Hence, until the laws of thermodynamics are repealed, to use Paul Samuelson's phrase,[43] it

[40] There are also signs that part of the slowness of admission may be due to administrative tardiness, as the blog posts by two Debian developers, Winnertz (2008, April 16) and Nussbaum (2008, April 15), suggest.

[41] See for instance HP's "Open Source and Linux from HP" web page (HP) and the Debian project's "Who's using Debian" web page (Debian.org).

[42] What constitutes good leadership is of course a topic in its own right. A pertinent study in this regard is O'Mahony and Ferraro (2007). This study focuses on the Debian project in an attempt to draw insights on leadership and governance which may be of relevance to communities engaged in the production and sharing of knowledge.

[43] Quoted in Mishra (2007, p. 15). The full quote is, "Until the laws of thermodynamics

would be futile to expect to get something for nothing, or outputs without inputs. Closed systems will also continue to be liable to decline and breakdown because they have to rely on endogenous or internally available inputs, which are finite in supply. Open systems, on the other hand, are able to draw on inputs from their environment, and such exogenous inputs, while not necessarily limitless in supply, are relatively abundant. This means that, barring any malfunctions in the system or adverse changes in the environment, open systems can keep going indefinitely.

Based on our taxonomy of GNU/Linux distributions, in which we classify distributions into closed system distributions and open system distributions, we also argue that while there may be nothing inherently wrong with the current proliferation of distributions, a phenomenon we describe as “widening”, there is also a need for “deepening”, which refers to the need for sustained improvements to the GNU/Linux system. Though we admit that there are undoubtedly many skilled and ingenious developers working on closed system distros, in our view, given their greater access to economic resources and a much larger pool of skilled developers, open systems are in a much better position than closed systems to produce fundamental improvements to the GNU/Linux system.

10 Maddog’s Epiphany.

Jon "Maddog" Hall is a well known personality in GNU/Linux circles.[44] During an interview on FLOSS Weekly (TWiT, 2007), a popular free and open source software podcast, Hall recounted the following story. Back in the mid-1990’s, when he was a marketing executive with the now-defunct Digital Equipment Corporation (DEC), Hall was on his way back home from a trip to

are repealed, I shall continue to relate outputs to inputs - i.e. to believe in production functions.”

[44] Maddog Hall is currently Executive Director of Linux International, a non-profit organization of computer vendors who support and promote the GNU/Linux system.

Australia, when he made a stopover in Fiji and checked into a resort called the Hideaway Resort hoping to enjoy a brief vacation. But the Hideaway couldn't have been much of a hideaway because the local DEC salesman found him and managed to persuade him to give a talk on the GNU/Linux system at the local university. At the university, Hall soon discovered that although the professors knew about GNU/Linux, it was difficult for them to get hold of a copy because at the time Fiji had an extremely dodgy connection to the Internet and every time anyone tried to download "a copy of the [Linux] kernel...they'd get it halfway through and some storm in the South Pacific would hit and drop the line."

So Hall took out the last GNU/Linux CD he had with him and as he handed it to the professors he was suddenly reminded of the painting in the Sistine Chapel in which God is giving the touch of intelligence to Adam.[45] And this was when the penny dropped for him. Up until this point, he had been more impressed by the low or zero cost of the GNU/Linux system, but as he handed his last CD over it suddenly struck him that in that CD was the equivalent of billions of dollars of ideas and work that had been contributed by people from all over the world who were motivated by a desire to build a better operating system. "And that's when I really begun to understand the whole thing," Hall confesses in the interview.[46]

While we might have mixed feelings about Richard Stallman's more radical claims regarding the immorality of non-free software,[47] we readily acknowledge that free software is a Promethean gift to the world as vividly

[45] The painting is Michelangelo's Creation of Adam. An image of the painting can be found at Wikipedia (Wikipedia-c).

[46] The story of Maddog's Fijian encounter doesn't end there, but we shall leave it to the reader to find out what happened next. Actually, the entire interview is well worth a listen, not only for Maddog's entertaining anecdotes, but also for the insider's perspective he provides on the computing industry and the early years of GNU/Linux.

[47] On strictly philosophical grounds, it is possible to argue that while free software may be the ethically superior alternative, this does not by itself make non-free software unethical (notwithstanding the fact that a certain prominent non-free software company has been known to behave in an ethically questionable manner).

portrayed in Maddog Hall's story. We also acknowledge that without Stallman's unwavering conviction and dogged determination, free software as we know it today might not exist. However, at the microeconomic level, how to ensure the sustainability of free software remains a vexed question.

As we have explained, in a market economy goods will be produced when only those who paid for them can consume their benefits. However, when benefits accrue to people other than the buyer of the good the producer will have less incentive to produce more goods. In addition, the producer will also seek to protect any knowledge that provides an advantage over competitors by preventing the dissemination of such knowledge. This forms the crux of the software dialectic. Free software favours inclusion, sharing, and openness rather than exclusion, competition, and secretiveness, but for the market to provide goods it must be possible to exclude "free riders" from sharing in the benefits of a good and competitors from sharing in the benefits of vital knowledge.

In this essay we have discussed two approaches to the production of free software. These approaches are the commercial approach and the community approach. In the former approach, the software dialectic may be resolved by moving up the value chain, while in the latter approach the software dialectic may be avoided altogether by relying on donations and volunteers.

Though some scepticism still remains about the long-term viability of the free software model, the fact that free software has continued to grow from strength to strength has given many observers pause for thought, and many are now less likely to ridicule free software as a quixotic idea.

A popular video on the Red Hat website quotes the Indian political and spiritual leader Mohandas Gandhi: "First they ignore you, then they laugh at you, then they fight you, then you win."[48] Whether the free (and open source) software model will emerge as the dominant software development

[48] The name of the video is "Truth Happens." It can be viewed or downloaded on the Red Hat "Videos" web page (Red Hat-c).

model, or whether this is just hyperbole, is something that remains to be seen. Even so, it is clearly not a possibility that can be dismissed out of hand.

About the Author: Mike Chege is an economist based in Nairobi, Kenya. His firm, Carpe Diem Africa, performs economic research and also assists small and medium-sized enterprises in business planning and marketing strategy. Mike is also a commentator and advocate on matters relating to information and communication technologies (ICTs) in less-developed countries. Comments on this paper would be appreciated and can be sent to `mikechege@gmail.com`.

Acknowledgments: Thanks are due to Dr. Nginya Lenneiye (World Bank) for suggesting the link between the Ubuntu philosophy and communitarianism, and Daniel Chable (former Chief Actuary, Nestlé) for his helpful comments on the paper, and in particular the entropy concept. The usual disclaimer applies.

References

[1] Amor, J.J., Robles, G., Gonzalez-Barahona, J.M., Pena, J. F. S. (2007). *Measuring Etch: The size of Debian 4.0.* GsyC/Libresoft, at `https://penta.debconf.org/~joerg/attachments/33-measuring_etch_slides.pdf`.

[2] Baumgärtner, S. (2004). "Thermodynamic models." In Proops, J. & Safonov, P. , (Eds.), *Modelling in ecological economics*, pp. 102-129, at `http://www.eco.uni-heidelberg.de/ng-oeoe/research/papers/Baumgaertner2004ModEE.pdf`.

[3] Berlecon Research. (2002). *FLOSS final report-Part 3. Free/Libre Open Source software: Survey and study. Basics of open source software*

markets and business models. Berlin: Berlecon Research, at http://www.berlecon.de/studien/downloads/200207FLOSS_Basics.pdf.

[4] B. Perens, 2005. "The emerging economic paradigm of open source." *First Monday, Special Issue #2: Open Source*, October, 2005, at http://firstmonday.org/issues/special10_10/perens/index.html.

[5] Cohen, A.J. & Harcourt G.C. (2003). "Whatever happened to the Cambridge capital theory controversies?" *Journal of Economic Perspectives,* Volume 17, Number 1, Winter 2003, Pages 199?214, at http://www.econ.yorku.ca/~avicohen/Linked_Documents/JEP_Cohen_Harcourt.pdf.

[6] Corning, .P.A. (2002) *Thermoeconomics: Beyond the second law.* Institute for the Study of Complex Systems, at http://www.complexsystems.org/publications/pdf/thermoecon.pdf.

[7] Coughlan, S. (2006, September 28). "All you need is Ubuntu." *BBC News Magazine*, at http://news.bbc.co.uk/1/hi/magazine/5388182.stm.

[8] Daly, H. & Cobb, J. (1994). *For the common good.* Boston, MA: Beacon Press.

[9] Danzig, T. (2005, November 25). "Ping," at http://taldanzig.blogspot.com/2005/11/ping.html.

[10] Debian.net, "Status of maintainer's packages, ports and bugs," at http://asdfasdf.debian.net/~tar/bugstats/?8.

[11] Debian.org, "Who's using Debian?" at http://www.debian.org/users/.

[12] Debian Times, "Debian GNU Linux powers Max Planck Institute 32.8 TFlops supercomputer," at http://times.debian.net/#1250.

[13] DeLong, J.V. (2004). *The enigma of open source software (Version 1.0)*. The Progress & Freedom Foundation, at http://www.pff.org/issues-pubs/pops/pop11.8opensource.pdf.

[14] DieOff, "Energy, entropy, economics, and ecology," at http://dieoff.org/page17.htm.

[15] Distrowatch-a, http://www.distrowatch.com.

[16] Distrowatch-b, "Search distributions," at http://distrowatch.com/search.php?status=All.

[17] Garbee, B. (2007) "Hewlett Packard's relationship with Debian." *DebConf7*, June 17-23, Edinburgh, Scotland, at http://meetings-archive.debian.net/pub/debian-meetings/2007/debconf7/low/077_Hewlett_Packards_Relationship_with_Debian.ogg.

[18] Gay, J., (Ed.), 2002. *Free software, free society: Selected essays of Richard M. Stallman*. Boston, MA: GNU Press. Available online at http://www.gnu.org/philosophy/fsfs/rms-essays.pdf.

[19] Georgescu-Roegen, N. (1971). *The entropy law and the economic process.* Cambridge, Mass: Harvard University Press.

[20] Gowdy, J. & Mesner, S. (1998). "The evolution of Georgescu-Roegen's Bioeconomics." *Review of Social Economy Vol LVI No. 2*, pp. 136-156, at http://www.rpi.edu/~gowdyj/mypapers/RSE1998.pdf.

[21] Heylighen, F. & Campbell, D.T. (1995). "Selection of organization at the social level: Obstacles and facilitators of metasystem transitions." *World Futures: The journal of general evolution*, Vol. 45, 1-4, at ftp://ftp.vub.ac.be/pub/projects/Principia_Cybernetica/WF-issue/Social_MST.txt.

[22] Hirsch, F. (1978). *Social limits to growth.* London: Routledge and Kegan.

[23] HP, "Open Source and Linux from HP," at `www.hp.com/go/debian`.

[24] Knopper.net,"ADRIANE - Audio Desktop Reference Implementation and Networking Environment," at `http://www.knopper.net/knoppix-adriane/index-en.html`.

[25] Joslyn, C. (1990, August). *On the Semantics of Entropy Measures of Emergent Phenomena*, at `ftp://ftp.vub.ac.be/pub/projects/Principia_Cybernetica/Papers_Joslyn/Entropy_Semantics.tex`.

[26] Krafft, M.F. (2005). *The Debian system: Concepts and techniques.* Germany: Open Source Press GmbH.

[27] Lakhani, K.R. & Wolf, R.G. (2005). "Why hackers do what they do: Understanding motivation and effort in free/open source software projects." In Feller, J., Fitzgerald, B. & Hissam, S., & Lakhani, K.R. *Perspectives on free and open source software.* MIT Press. Available online at `http://freesoftware.mit.edu/papers/lakhaniwolf.pdf`.

[28] Lerner, J. & Tirloe, J. (2000). *The simple economics of open source.* Harvard Business School and NBER, at `http://www.people.hbs.edu/jlerner/simple.pdf`.

[29] LWN.net, "The LWN.net Linux distribution list," at `http://lwn.net/Distributions/`.

[30] Mandriva. (2006, March). "Interview of François Bancilhon, CEO of Mandriva," at `http://club.mandriva.com/xwiki/bin/view/Main/interview-fbancilhon-2006-03`.

[31] Mandriva-a, "Backgrounder," at `http://www.mandriva.com/enterprise/en/company/backgrounder`.

[32] Martin, R. (2007, November 20). "Bob Young and the rise of Red Hat software." *The Globe and Mail*, at http://www.reportonbusiness.com/servlet/story/RTGAM.20071120.wmartin1121/BNStory/robAtWork/home.

[33] McGregor, C. (2008, January 23). "Interview with Richard M. Stallman." *Free Software Magazine*, at http://www.freesoftwaremagazine.com/node/2564/pdf.

[34] Michlmayr, M. (2004). "Managing volunteer activity in free software projects." *Proceedings of the FREENIX Track: 2004 USENIX Annual Technical Conference*, June 27-July 2, 2004, Boston, MA., at http://www.cyrius.com/publications/michlmayr-mia.pdf.

[35] Mishra, S.K. (2007). *A brief history of production functions.* MPRA Paper No. 5254. Munich Personal RePEc Archive, at http://mpra.ub.uni-muenchen.de/5254/1/MPRA_paper_5254.pdf.

[36] Murdock, I. (1993, August 16). "New release under development; suggestions requested," at http://groups.google.com/group/comp.os.linux.development/msg/a32d4e2ef3bcdcc6.

[37] _________. (1994, June 1). *The Debian Manifesto*, at http://www.debian.org/doc/manuals/project-history/ap-manifesto.en.html.

[38] Musgrave, R. A. & Musgrave, P. B. (1989). *Public finance in theory and practice.* New York: McGraw Hill.

[39] Nussbaum, L. (2008, April, 15). "4 months and 10 days without any new Debian developer. Is Debian dying?" at http://www.lucas-nussbaum.net/blog/?p=286.

[40] O'Mahony, S. & Ferraro, F. (2007). *The emergence of governance in an open source community.* Technology, Innovation

and Institutions Working Paper Series. University of Alberta, at http://www.techforce.com.br/index.php/news/linux_blog/scientific_study_about_debian_governance_and_organization.

[41] Raymond, E. (1999). *The cathedral and the bazaar.* Sebastopol, CA: O'Reilly. Available online at http://www.catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar/.

[42] _______. (1999). "The revenge of the hackers." In DiBona, C., Ockman, S., & Stone, M., (Eds.), *Open sources: Voices from the open source revolution.* Sebastopol, CA: O'Reilly & Associates. Available online at http://www.oreilly.com/catalog/opensources/book/raymond2.html.

[43] Reason, T. (2001, September 13). "Linux: When free isn't," cfo.com. Cited in Smith, B.L. "The future of software: Enabling the marketplace to decide," p. 466. In Wynants, M. & Cornelis, J., (Eds.), *How open is the future? Economic, social & cultural scenarios inspired by free & open-source software.* Brussels: VUBPRESS. Available online at http://crosstalks.vub.ac.be/publications/howopenisthefuture/crosstalks_book1.pdf.

[44] Red Hat. (2007b). Red Hat Summit 2007, minute 00.27, at http://www.redhat.com/videos/summit2007/.

[45] _______. (2008). Form 10K Red Hat Inc., at http://media.corporate-ir.net/media_files/IROL/67/67156/REDHATINC10K2008.pdf.

[46] Red Hat-a, "Open source development list," at http://www.redhat.com/truthhappens/leadership/osdevelopment/.

[47] Red Hat-b, "Investor relations," at http://phx.corporate-ir.net/phoenix.zhtml?c=67156&p=irol-irhome.

[48] Red Hat-c, "Red Hat videos," at http://www.redhat.com/videos/ourfilms.html.

[49] Robinson, J. & Eatwell, J. (1973). *An introduction to modern economics.* Great Britain: McGraw Hill.

[50] Robles, G., Gonzalez-Barahona, J.M., Michlmayr, M. (2005). "Evolution of volunteer participation in Libre Software projects: Evidence from Debian." Scotto, M., & Succi, G., (Eds.), *Proceedings of the First International Conference on Open Source Systems*, pp. 100-107, at http://www.cyrius.com/publications/robles_barahona_michlmayr-evolution_participation.pdf.

[51] Rossi, M.A. (2004). *Decoding the free/open source (f/oss) puzzle.* Dipartimento di Economia Politica, Università degli Studi di Siena, at http://opensource.mit.edu/papers/rossi.pdf.

[52] Rullani, F. (2005). *The three dimensions of a communitarian institution: The open source software community case.* Working Paper No. 16. Faculty of Economics, University of Bologna, at http://www.aiccon.it/file/convdoc/n.16.pdf.

[53] Schrödinger, E. (1945) *What is life? The physical aspect of the living cell.* New York: Macmillan.

[54] Stavrianos, L.S. (1990). *Lifelines from our past.* London: I.B. Tauris.

[55] Stoveland, J.F. (2008). *Managing firm-sponsored open source communities: A case study of Novell and the openSUSE project.* Master's Thesis. Department of Informatics, University of Oslo, at http://janfredrik.wordpress.com/master-thesis/.

[56] TLLTS (2006, October 25). Show No. 164, minute 68:37, at http://tllts.org/mirror.php?fname=tllts_164-10-25-06.ogg.

[57] ______. (2007, April 11). Show No. 187, minute 84:00, at http://tllts.org/mirror.php?fname=tllts_187-04-11-07.mp3

[58] TWiT. (2007, March 24). FLOSS Weekly 17: maddog, minute 53:00, at http://www.podtrac.com/pts/redirect.mp3/twit.cachefly.net/FLOSS-017.mp3.

[59] Vaughan-Nichols, S.J. (2005, September 29). "There are too darned many Linuxes," at http://www.linux-watch.com/news/NS8823760499.html.

[60] _______. (2007, November 1). "No worries: Head of MEPIS Linux is fine," at http://www.desktoplinux.com/news/NS7389992145.html.

[61] Wikipedia-a, "Knoppix," at http://en.wikipedia.org/wiki/Knoppix.

[62] Wikipedia-b, "Klaus Knopper," at http://en.wikipedia.org/wiki/Klaus_Knopper.

[63] Wikipedia-c, "Creation of Adam," at http://upload.wikimedia.org/wikipedia/commons/7/73/God2-Sistine_Chapel.png.

[64] Winnertz, P. (2008, April 16). "Demotivation of new prospective developers," at http://www.der-winnie.de/posts/Demotivaton_of_new_prospective_Developers/.

[65] Xulu, M. (2007, November). "Ubuntu is learned, not natural or automatic, behavior," *ANC Today, Volume 7 No. 43*, November, 2007, at http://www.anc.org.za/ancdocs/anctoday/2007/at43.htm#art1.

[66] Yahoo!Finances, "Red Hat Inc.," at http://finance.yahoo.com/q?s=RHT.

[67] Young, B. (1999). “Giving it away: How Red Hat stumbled across a new economic model and helped improve an industry.” In DiBona, C., Ockman, S., & Stone, M, (Eds.), *Open sources: Voices from the open source revolution.* Sebastopol, CA: O’Reilly & Associates. Available online at `http://www.oreilly.com/catalog/opensources/book/young.html`.