

**'LIBRE' SOFTWARE: TURNING FADS INTO INSTITUTIONS?**

**Jean-Michel Dalle**

Ecole Normale Supérieure de Cachan  
61, avenue du Président Wilson - F-94230 Cachan  
Tel: (33) (0)49085995 - Fax: (33) (0)149085996  
e-mail: < jmdalle@dir.ens-cachan.fr >

**Nicolas Jullien**

ENST Bretagne & ICI  
Technopole de Brest Iroise - F-29285 Brest Cedex  
Tel: (33) (0)298001245 - Fax: (33) (0)298001173  
e-mail: < Nicolas.Jullien@enst-bretagne.fr >

**ACKNOWLEDGEMENTS:** The authors wish to thank Mélanie Clément-Fontaine, Godefroy Dang Nguyen, Laurent Kott, Morris Teubal and Jean-Benoît Zimmermann for many helpful discussions.

**ABSTRACT:** The article presents an economic analysis of Libre software and of its sustainability as an economic model. We underline the role of Libre software development communities and analyze incentives of both kernel and obscure developers. We specially emphasize the role of the so-called 'public' licenses to provide an appropriate institutional framework. We show that several features of Libre software also allow it to improve faster than proprietary software, and therefore to achieve strong market performance when competing against existing standards, even when proprietary software producers react. We illustrate our point using a simple local and global interaction model to study the technological competition between Linux and Windows on the server operating system market. We finally argue that Libre software could turn from a fad into an efficient economic institution under certain conditions i.e. if developer communities could be created with appropriate incentive structures and if sufficient initial momentum could foster its diffusion without relying on purely cultural feelings such as anti-Microsoft attitudes.

**KEY WORDS:** Libre software, Linux, Community, Incentives, Network Effects, Technological Competition.

## 1. 'Libre' software

What should we call software distributed with its sources *and* with the right to modify and redistribute it as long as it retains the same characteristics, e.g. software like Linux, the emerging new star in the operating systems (OS) market? Such software is often referred to as 'open-source' software or as 'free' software, the first expression being now somewhat more frequent than the second. Unfortunately, both expressions are misleading and somewhat inadequate. First, Linux is not necessarily free, and is certainly not going to be free for most users, as distributions of Linux are now being *sold* at classical retail stores by various companies (Red Hat, Correll, Süse, etc.) while many other companies are selling costly *services* to Linux users for them to truly benefit from their new OS. Second, there are many examples of open-source software which is indeed *proprietary* and which certainly does not qualify as being the kind of software which we are considering, since openness of sources does not guarantee that it can be modified easily by anyone, and certainly not that anyone is allowed to 'freely' redistribute it. Many business firms are indeed currently trying to make their software products 'like' Linux, in an attempt to copy Linux' success: for the moment, they have largely failed, or at least have not succeeded, as they are mostly opening sources to attract external contributors who would participate for free in the development of their product, but without granting these contributors the secure development context that Linux provides. As a consequence, Linux is neither 'free' software nor simply 'open-source' software. Following a recent report to the European Commission, we would like to suggest calling Linux and its fellows 'Libre' software (with a capital L to avoid misunderstanding): in French, 'Libre' refers to liberty and to freedom and not to gratuity<sup>1</sup>. Libre software is software distributed with its sources *and* with the right to modify it and to redistribute it as soon as it remains Libre.

So Linux is a piece of Libre software, and Libre software is gradually appearing as one of the most fashionable and possibly one of the most interesting new economic models in today's software industry. It indeed allows users to cooperate – essentially through the Internet – by making most of the time marginal improvements to a given piece of software before it is redistributed once modified. In this way, each user rapidly benefits from innovations brought by all others. Libre software is thus a very seductive concept, but all the interest it has attracted has not only been theoretical since it has also much to do with pure commercial success. Linux, the most well-known example of Libre software, has clearly appeared as a major challenge to Microsoft OS Windows NT and now 2000<sup>2</sup>: it is reported to run more than 25% of Internet servers<sup>3</sup> and its commercial shipments are seen by some analysts to be growing faster than any other operating software (25% vs. 10% per year<sup>4</sup>). Meanwhile, the web server "Apache" is leading its market with no less than a 60% market share<sup>5</sup> -, and the mail server "Sendmail" unambiguously dominates its own market, not forgetting the programming language "Perl" nor the many other examples.

---

<sup>1</sup> It would certainly not be the first time that either English or French borrows words from each other's language when they are clearly more appropriate.

<sup>2</sup> In a recent Microsoft internal draft, Libre software has been characterized as a « new development methodology », and analyzed as a major challenge to Microsoft products.

<sup>3</sup> <http://leb.net/hzo/ioscount/>

<sup>4</sup> <http://www.idc.com/Data/software/content/SW033199PR.htm>

<sup>5</sup> <http://www.netcraft.co.uk/Survey/>

Such success is of course challenging for economists, and specially so since Libre software was truly born much more as a fad – mainly rooted in anti-proprietary software and more specifically in anti-Microsoft and in anti-Windows attitudes – than as a sustainable economic model or as a relevant economic institution. Here comes then the major economic question this article is attempting to address: will Libre software be able to turn from a fad into an economic model and an economic institution? We believe the answer is twofold.

First, we have to understand how Libre software ‘works’, which notably refers to the incentives issue (section 2). It is indeed extremely puzzling for classical economic theory that developers choose to participate in Libre software development whereas they seemingly get no reward from doing so, and this very issue is currently subject to serious economic analysis, notably by Josh Lerner & Jean Tirole (Lerner & Tirole, 2000) and by Von Hippel and colleagues (Lakhani & Von Hippel, 2000; Harhoff, Henkel & Von Hippel, 2000). In this context, we argue here that several incentives structures co-exist and apply differently to *heterogeneous* developers. Career concerns – critically enhanced by the emergence of ancillary service companies created by or with key members from the developer community (Dalle & Jullien, 2000) – clearly provide an appropriate answer to the incentive issue, and therefore allow Lerner & Tirole (2000) to emphasize “*simple* economics of open-source”: however, career concerns apply only to *kernel* developers, as they seem to apply only to a limited subset of individuals and organizations<sup>6</sup>. For others, i.e. for *obscure* developers, contributing for instance only with a few lines of code for the Linux driver of an unusual printer, potential explanations notably include the low competitive value of such work and related knowledge, specially when provided by users who are also innovators (von Hippel, 1988).

But incentives are only part of the Libre software issue, and might indeed be the *simplest* part of the economics of open-source and Libre software. We argue here that Libre software depends upon a *community* of heterogeneous developers with its associated *institutions*: specially, no appropriate incentives would exist without a credible commitment framework as it is provided by “*public*” licenses like the General Public License (GPL). Public licenses indeed guarantee that Libre software protected by GPL will always remain Libre while it is gradually diffused and improved. Neglecting the role of public licenses and of the framework they provide to developer communities has indeed led most business firms to fail when they have tried to implement features of Libre software in their business model. Incentives to work for free are simply not sustainable without an appropriate institutional framework and economic model. According to us, Libre software points at a *new economic and institutional model for creative (epistemic) communities*, which should be compared both with patents and with the conventions of Open Science in the scientific community.

Finally, Libre software not only ‘works’, but also works well, and we do not either have any satisfactory explanation yet for Libre software market performance (section 3), although such performance is both critical for the sustainability of the associated economic model and extremely puzzling for an economist, as Lerner and Tirole (2000) recognize<sup>7</sup>. We argue here that Libre software improves *faster* than proprietary software due to several characteristic factors of the Libre

---

<sup>6</sup> <http://orbiten.org/ofss/01.html>

<sup>7</sup> “Aspects of the future of open source development process, however, remain somewhat difficult to predict with ‘off-the-shelf’ economic models.” (Lerner & Tirole, 2000, p. 1).

software development model, namely higher increasing returns associated with adoptions by users who are also innovators, and a more efficient redistribution of these returns back to users through faster and more frequent release strategies. Taking as an example the battle between Linux and Windows on the market for server operating systems, we show with a simple stochastic interaction competition model that this ‘faster improvement’ property is a good candidate to explain why Libre software is able to defeat dominant proprietary standards, even when proprietary software competitors react quickly and strongly. Both stronger local externalities are necessary for this result to obtain, at least at the beginning of diffusion processes. We therefore conclude also that local proselytism favoring Linux in the developer community has played an essential role by creating sufficient *initial momentum*.

Both aspects of the Libre software economic model are finally relevant to question the sustainability of the Libre software development model. To be sustainable, Libre software clearly needs creative communities with both kernel and obscure developers for whom Libre software provides appropriate incentives as soon as their efforts are secured by GPL. This is why many private companies which tend to replicate the Libre software development model try to create dedicated communities by hiring kernel developers and to issue credible public licenses, although they have often failed for now in this last respect. These communities indeed account for most of Libre software market performance: but to be sustainable, Libre software also needs initial momentum which for now has mainly been provided by various kinds of proselytism. If Libre software was to be proved as improving collective welfare, for instance by counterbalancing monopoly trends in markets dominated by network effects, then some kind of public intervention might be needed not only to help create Libre software communities but also that the initial momentum which gives rise to Libre software diffusion should be recreated by means other than purely cultural feelings. To sum up, we conclude that Libre software might very well be turning into an efficient institution, and that further research is therefore needed, and obviously needed fairly soon, to determine where and when – for which kind of software, notably – such an institution would be efficient and should be encouraged, and when it should not.

## **2. The economics of Libre software communities**

### **2.1 Incentives: simple explanations**

Why do Libre software developers disclose proprietary information i.e. why do they contribute for free to Libre software development? A simple explanation lies in reputation effects and associated expected gains or else in signaling effects and therefore in career concerns (Dalle & Jullien, 2000; Lerner & Tirole, 2000). Both explanations are indeed similar and essentially build upon the existence of what we have called ‘ancillary business companies’, i.e. companies providing services to Libre software users, be they individuals or organizations, which are not directly supported by the development community (like commercial packaging, after-sale support or simply maintenance and customization of installed systems, etc.). These companies are now very common: at least a few of them exist for almost all major Libre software, most of which have been created by or in association with *kernel* Libre software developers (such as Linus Torvalds and Allan Cox for the Red Hat company). They very regularly offer job and even equity positions to kernel developers and therefore contribute to

creating rational expectations about the ability of kernel developers to transform into profit their reputation acquired while contributing to Libre software development: lines of code are « signed » by their authors, and knowledge of who did what is indeed widely accessible in the development community. Although such incentives are clearly targeted at kernel developers, they also partly apply to obscure developers. Contributing to obscure aspects of software development can indeed open the way for subsequent kernel developments, either in the same or in another Libre software community, while contributions, be they small, directly represent a positive signal for many job applications. However, associated profit expectations are to be rather weak as there is clearly a relatively poor reputation associated with the programming of hundreds of lines of relatively uninteresting code.

For obscure developers, a weak expected reward associated with disclosure is also counterbalanced by low creation costs – as long as we are considering *innovative users* (Von Hippel, 1988), as innovative users often develop obscure pieces of code for their own interest – and also by a very low reward associated with appropriation (non-disclosure), as obscure work has indeed almost no competitive ‘tradable’ value. Furthermore, similar solutions are also certainly to be proposed by other members of the community, generally making them, or at least some of them, better off if they disclose<sup>8</sup>. Think for instance of a developer who happens to use a rather rare printer: considering that he is able to develop the associated Linux driver, or to adapt an existing driver whose source code is in fact open and available, why should he *not* disclose his work? Considering further that they are numerous such developers, *and that they are heterogeneous*, it is finally nearly sure that at least one of them at least will contribute to that particular piece of Libre software development. And there are, indeed, many different such contributions in major Libre software communities among which the best is selected, hence improving efficiency further.

## **2.2 Incentives: less simple explanations**

But these two explanations, simple as they are, also rely on the existence of *credible commitment* mechanisms, provided by a key feature of Libre software i.e. the so-called public licenses or GPL (general public licenses), the best known of which is the GNU license<sup>9</sup>, generally referred to as the « CopyLeft » protection scheme. Incentives to disclose knowledge would indeed obviously be much weaker if someone was able to appropriate the associated reward, be it through profit or through reputation. All Libre software developers are reasonably sure that their CopyLeft *protected* work is to remain non-proprietary, as CopyLeft means that it is to be freely read, modified and redistributed but it certainly does not allow anyone to use it in any proprietary closed form<sup>10</sup>. GPL licenses guarantee that Libre software is actually not only open-source, but precisely *Libre*.

In fact, economic creativity, being highly subject to positive externalities, generally implies some kind of an associated institutional and/or conventional enforcement mechanism: since knowledge disclosure is a key condition of further

---

<sup>8</sup> See also Harhoff, Henkel & Von Hippel (2000) and Lakhani & Von Hippel (2000) for similar inquiries. Issues à la Kahneman & Tversky would certainly also be interesting to investigate in that respect.

<sup>9</sup> <http://www.fsf.org/copyleft/>. GNU is a « recursive » acronym for GNU's Not Unix.

<sup>10</sup> Although there has been no trial yet, and although some aspects of public licenses remain questionable: see also <http://crao.net/gpl>.

knowledge improvement – knowledge is essentially built upon previous knowledge –, the social value of creating and disclosing knowledge is significantly higher than its individually appropriable value and there is a need for well-adapted institutional and/or conventional devices, which of course often rely on customized property regimes (David & Foray, 1995). This is both the case for patents and for the scientific community, to name but two major examples. Patents as an institutional device correct this by granting patentees – who have to publish their discovery – with temporary monopoly power, with possible enforcement by law courts. In the scientific community, a similar mechanism is provided by conventional means: reputation is granted to scientists who happen to be the first to publish results (Dasgupta & David, 1994). Historically closer to the scientific community (Jullien, 1999), Libre software fundamentally appears as a kind of an *anti-patent* device, as property is not granted but denied. Denying appropriation paradoxically creates incentives towards disclosure as induced spillovers will neither be appropriated by the inventor *nor by others*.

		Patents	Open-Science	Libre software
<b>Creation and Disclosure</b>	<i>Incentives</i>	Monopoly power	Reputation by peer recognition	Reputation, signaling, career effect and induced profit
	<i>Enforcement</i>	Law courts via infringement trials	Conventional, self regulation of the research community	CopyLeft and GPL licenses as anti-patent devices
<b>Distribution</b>	<i>Devices for information exchange and pooling</i>	Patent databases maintained by public agencies	Scientific journals, Conferences, email	Internet (email, mailing lists, web sites, newsgroups)

Table 1: Different institutional devices dealing with creativity.

Public licenses have proved to be an appropriate tool for transforming communities of practices – of users – into epistemic – creative – communities (Cohendet, Créplet & Dupouët, 2000). As an immediate corollary – and as an indirect proof of what we are stating here –, we are also provided with a straightforward explanation for the relatively general failure of business firms, such as Netscape, which have tried to implement open-source rather than Libre features in their previously proprietary development model. They have been unsuccessful in motivating large and creative enough Libre software communities. Even when they hire kernel developers, or manage to be publicly supported by a few of them, they are unable to attract sufficient numbers of other developers, notably obscure ones, as long as they try to cling to not-completely-public, and therefore somewhat-still-proprietary, licenses. Many such attempts have aborted due to inappropriate licensing schemes, most of the time due to actual doubts by potential developers about the credibility of the commitment by the applicant firm to a Libre software development model. This is indeed an important lesson for all proprietary software producers interested in fostering Libre software processes associated with some of their products<sup>11</sup>.

---

<sup>11</sup> As a matter of fact, the ability of would-be ancillary companies to deal with Libre software communities is to be a key condition of their success: they have to deal with a

This last issue is all the more important as hybridization (Dalle, Jullien & Simon, 2000) between Libre software communities and business firms is a candidate as an explanation for the ever-increasing efficiency of Libre software. Firms essentially deal with what Libre software communities cannot do – they sort of ‘supplement’ Libre software communities – while earning money from selling associated services. They notably contribute to the development of obscure parts of Libre software and therefore considerably improve the user-friendliness of Libre software *products*. They also support a better coordination of Libre software projects: as a matter of fact, although many Libre software communities have been able to implement sometimes quite amazing organization schemes and decision rules – this is notably the case for the Linux community – there is still some doubt about their general ability in this respect, notably because of the risk of ‘code-forking’ which refers to the eventuality of a Libre software community splitting into two or more alternative sub-communities developing similar but increasingly different and potentially incompatible software products. However, a similar risk exists with ancillary companies as they have incentives to make Libre software they distribute specific and thus somehow proprietary again.

In any case, it should be noted as a conclusion to this section that no reference has been made here to anti-Microsoft feelings or even to purely cultural issues regarding the involvement of developers into Libre software communities. In fact, Linux is just one particular example of Libre software, and recent and older Libre software seems to fit even better into the framework we have outlined here, though attention should certainly be paid to the fact that some of these issues apply differently depending on software products. However as it is still under construction, the Libre software model therefore appears as perfectly sustainable as far as incentives are concerned. We have now to tackle the other issue about Libre software market performances against proprietary software, since the actual role of Libre software in the software economy will also depend on its ability to win competitive situations against proprietary products. In a way, incentives to be part of a Libre software community also considerably depend on the expectations about Libre software market performance: if it were poorer, incentives would obviously be much lower.

### ***3. An assessment of Libre software efficiency and market performance***

#### ***3.1 Libre software improves faster***

---

community of « geeks », to quote a word now fashionable even among geeks themselves. This topic was indeed listed among main risk factors in a recent Red Hat IPO prospectus: « Negative reaction within the open source community to our business strategy could harm our reputation and business [...]. [Some] have suggested that [...] we are trying to dominate the market for Linux-based operating systems and the open source community [...]. This type of reaction, if widely [...], could harm our reputation, diminish the Red Hat brand and result in decreased revenue.» Creating creative enough communities – attracting both kernel and obscure developers with associated dynamic club effects – is indeed generally a key issue here, and a few firms have thus begun to propose consultancy and technology services to other businesses willing to do so.

As Eric S. Raymond<sup>12</sup>, a major Libre software advocate, once put it: « From nearly the beginning, [Linux] was rather casually hacked on by huge numbers of volunteers coordinating only through the Internet. Quality was maintained [... by the] strategy of releasing every week and getting feedback from hundreds of users within days. » As we have already outlined, the *efficiency* of Libre software indeed comes from the fact that it is supported by a community which benefits from extremely low development costs and from powerful communication technologies thanks the Internet, from emails to mailing lists and to Web servers. Innovation is thus easily decentralized (Cohen, 1983) and numerous developers are able to push and publish their ideas while only the best are selected, a feature which proprietary software producers have difficulties in coping with. Moreover, Libre software innovators also being users, Libre software programs are developed to cope with problems which users really face, therefore considerably improving their efficiency through much quicker and better development feed-back. As a comparison, proprietary software producers are instead often far removed from their clients and typically have to organize huge marketing studies to decipher their needs.

But following Eric S. Raymond, the way Libre software works is not only critical for its improvement, i.e. for bug fixing, it has also other considerable consequences on the way newer versions are *released*, which indeed creates another major difference between Libre and proprietary software. Proprietary software users usually have to pay for newer, although sometimes only slightly improved, versions, and often have to wait for a long time before appropriate patches are even released to fix even important bugs. On the contrary, Libre software improvements are regularly and rapidly accessible through the Internet. To summarize, lower development costs and better and quicker feed-back from a community of developers also result in very different *release strategies*. Therefore the pace of improvement of both proprietary and Libre software will be critically different not only because improvements cost less and are more efficient, but also because they are made accessible to users in more efficient ways.

To put it differently, Libre software generally appears as a more efficient way than proprietary software of dealing with positive external economies associated with technological development and diffusion. The result is a faster improvement of Libre software as compared to proprietary software. Libre software allows a more extensive redistribution to consumers of increasing returns associated with technological adoption not only because its diffusion generates higher increasing returns per se, but also because they are not appropriated by producer through opportunistic release strategies. This should remind us of a black box within Arthur's (1989) classical model of technological competition, according to which increasing returns of adoption are assumed to be re-invested in technological improvement by producers, i.e. in a way redistributed to consumers. In fact, proprietary software producers *choose* between profits and investments in further improvements of their products. They select release strategies according to their market power such that they allow higher margins and recurrent income, even when this is not perfectly satisfactory for consumers<sup>13</sup>. In other words, the rate of improvement of the utility of a technology depending on the extent of its diffusion is influenced by choices between alternative release strategies: as for Libre software, it will generally be structurally higher than for proprietary software.

---

<sup>12</sup> <http://www.tuxedo.org/~esr/>

<sup>13</sup> Not to speak of monopolists' strategies.

### 3.2. Linux vs. Windows

This last aspect is of course of considerable importance for the assessment of Libre software market performance in competitive situations. To further illustrate this point, we now turn to the competition between Linux and Windows in the server operating systems market. Apart from being fashionable, this situation is interesting for at least three main reasons: first of all, it concerns operating systems which are both at the core of any computer and highly subject to network effects associated with compatibility issues; secondly, it is a situation where a new entrant – Linux – is trying to invade a market already dominated by an existing standard – Windows –; and thirdly, Linux is also subject to strong positive local externalities due to the proselytism of Linux adopters: there are for instance numerous associations such as Lugs (Linux Users Groups) who organize regular events to promote its use, while Linux users generally tend to actively promote its use and to publicize its quality and interest to other potential users in their “neighborhoods” i.e. among their acquaintances. As a matter of fact, the ongoing success of Linux – which is continuously and consistently gaining market shares – would be very difficult to assess if we were only to consider global positive externalities, even under the hypothesis that Libre software improves faster than proprietary software, as Arthur and followers<sup>14</sup> have shown. To put it briefly, Arthur’s model neglects local interactions whereas technological adoption generally depends also on the previous choices of a subset of local ‘relevant’ neighbors (David, 1988).

We therefore turn to a now rather conventional local and global interaction model<sup>15</sup>. We consider a population of heterogeneous potential adopters with different decision rules. Depending on the adopters, quality, performance (frequency of errors), availability and variety of dedicated software, easiness of installation and comfort of use, direct and indirect costs (buying, training, maintenance, upgrades, dedicated software), will indeed be more or less relevant parameters to evaluate the utility of alternative software solutions. To give but one example, such valuations will indeed most certainly be very different from a computer hacker to an unskilled computer user in a major company. This is why we should turn to statistical analysis, simply accounting for the statistical propensity of a given adopter randomly chosen in a given population to adopt either one of the two competing standards.

Yet, individual valuations are also highly sensitive to both local and global externalities. As is now well-known, global characteristics of products, like quality, performance, availability and variety of dedicated software, price, depend more or less directly on their diffusion level. As a consequence, the statistical propensity of a random adopter to choose a given technology will also be a function of the market share of this technology and even more so in the case of network technologies for which compatibility issues play a major role. But these compatibility and externality issues are also specially prominent within the limited set of other adopters with whom a given adopter often interacts, i.e. *locally*: this is specially true for operating systems due to regular file exchange among users.

What we therefore have to study is the statistical propensity of a given population to adopt each technology conditional on “previous” global and local adoption patterns. And, as we have outlined above, both local and global effects are different for Libre and proprietary software: they will both be stronger for

---

<sup>14</sup> See e.g. Durlauf (1993), David & Foray (1994), David, Dalle & Foray (1998), Dalle (1995, 1997), Kirman (1993, 1998).

<sup>15</sup> See Dalle (1998ab) for this particular implementation.

Libre software. First, Linux (Libre software) improves more quickly than Windows (proprietary software), as evolutions of Linux are constantly accessible on-line and for free while Windows users have, except for important bugs and very minor improvements, to wait for years until a new version is available, and they have also to *buy* it<sup>16</sup>: as a consequence, global externalities are stronger<sup>17</sup>. Second, Linux users are more prone to proselytism than Windows users, as Linux users advertise the quality of their technology, while also stressing all the problems experienced by Windows: therefore local externalities are also stronger.

We therefore suggest following statistical demand function:

$$\mathbf{Prob}[Agent A \text{ adopts Linux}] = \frac{th\left[a\left((x_l - \alpha x_w) + b(X_l - \beta X_w)\right)\right] + 1}{2}$$

$x_l$  (resp.  $x_w$ ) is the proportion of A's neighbors who have adopted Linux (resp. Windows) while  $X_l$  (resp.  $X_w$ ) is the proportion of Linux (resp. Windows) adopters in the entire population.  $a$  estimates the (statistical) preference for standardization: the greater  $a$ , the more a given potential adopter is driven towards standardization because of compatibility issues;  $b$  stands as an estimate for the statistical preference for global standardization vs. a local one. Both  $a$  and  $b$  are characteristics of technologies and specifically of their associated *network effects*. Typically, both  $a$  and  $b$  will be high for operating systems such as Linux and Windows. In the limit case, technologies with no associated externalities would correspond to  $a = 0$ .  $\alpha$  statistically estimates the *relative* influence of previous local Windows adoptions as compared to previous local Linux adoptions;  $\beta$  statistically estimates the *relative* influence of previous global Windows adoptions as compared to previous global Linux adoptions. As local and global externalities are assumed to be higher for Linux than for Windows (see above), we have :

$$\alpha < 1 ; \beta < 1$$

i.e. a random potential adopter will adopt Linux more frequently than Windows when the number of previous adopters of Linux and Windows is the same either in its neighborhood or globally: the propensity to adopt Linux will be higher than the propensity to adopt Windows with a similar level of either local or global positive externalities. As for Windows,  $\alpha$  and  $\beta$  are specially influenced by the choice of regular release and sale of somewhat improved and imperfectly compatible versions and by a monopoly pricing strategy, i.e. in both cases by a monopolistic behavior regarding both quality and price of products. As for Linux, the higher the proportion of active users who agree to make use of their skills to correct bugs and improve Linux, the lower  $\beta$  as Linux adoptions will provide even more externalities. Similarly, proselytism for Linux – or conversely Microsoft-phobia – will influence  $\alpha$  since proselytism is mainly local. The existence Ancillary businesses such as Red Hat or VA Linux also lowers  $\beta$  as they notably provide increased user-friendliness and after-sales support. Finally, making use

---

<sup>16</sup> Microsoft, being a monopolist, has specially low incentives to re-invest increasing returns of adoption: an explanation perhaps for a *very* slow improvement rate and for very opportunistic release strategies, somewhat more so than for other software producers.

<sup>17</sup> We consider here that there are no more differences in access conditions for Linux and Windows thanks to the Internet and to Red Hat, COREL, and other Linux packages.

of a hyperbolic tangent function, adding 1 and dividing by 2 is just a question of normalization.

### 3.3. Results

Each potential adopter is associated with a local neighborhood (a subset of other potential adopters): we assume here for tractability reasons that neighborhoods are organized as in a 2-dimensional torus - the « interaction structure » -, i.e. that all adopters have 4 local relevant neighbors<sup>18</sup>. We simulate technological trajectories by choosing a random adopter at discrete times, i.e. a position on the interaction structure and an adoption behavior given local and global environment, according to the statistical demand function outlined above: we repeat the algorithm up to 100 000 times and we measure the (possibly infinite) time needed for Linux to reach a 70% market share. Initial conditions are a uniform adoption of Windows. We repeat the entire process for each  $\alpha$  and  $\beta$  between 0 and 1 with step 0.1. Figure 1 below gives simulation results for  $a = 2$  and  $b = 1$ , which seem appropriate values for a technology highly sensitive to standardization phenomena such as operating systems.

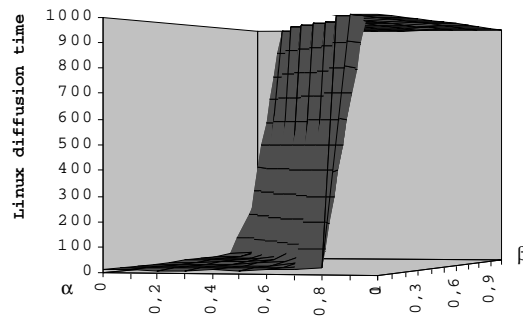


Figure 1: Linux diffusion time (static  $\alpha$  and  $\beta$  parameters)

These results show a phase transition, i.e. the fact that  $\alpha$  and  $\beta$  behave as state parameters associated with a sharp discontinuity in diffusion regimes. Above critical values of  $\alpha$  and  $\beta$  diffusion is infinitely long – it will never occur in economic time – whereas below them diffusion is almost sure and even fairly rapid. As a consequence, Linux wins when it benefits from significantly stronger global and local externalities than Windows<sup>19</sup>. Stronger global externalities

<sup>18</sup> Here for simulation simplicity a 30x30 2-dimensional torus (900 potential adopters).

<sup>19</sup> It is interesting to note here that a minor technological innovation might sometimes prove powerful enough to replace a dominant standard as long as it is associated with a superior economic model. Taken as a technology, Linux is indeed not really superior to Windows. Here superiority does not come from pure technological arguments, but from a superior *economic model*: non-technological aspects prove to be sufficient to help a non-superior technology to defeat a dominant standard. What characterizes technological trajectories is thus closer to persistence (Foray, 1997) rather than to pure and irreversible path-dependence (David, 1985): technological diffusion endogenously creates *endogenous thresholds*. Posterior technological candidates have to be « sufficiently better », be it technologically or economically, to get rid of existing standards, i.e. above a previously endogenously created threshold. As we argued elsewhere (Dalle, 1998b), this phenomenon offers a straightforward explanation to the

associated with the faster improvement of Libre software are therefore not sufficient. The diffusion of Linux is also highly sensitive to stronger local externalities i.e. to users' proselytic behavior.

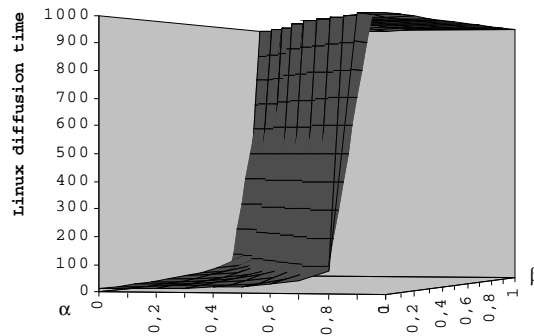


Figure 2: Linux diffusion time when  $\beta$  varies during diffusion

Very similar results obtain if we consider that Microsoft answers the Linux threat by lowering its prices or by changing its business model towards software renting instead of selling, i.e. by making a major shift in its release strategy however difficult this might prove for as big a company as Microsoft, deeply engaged in long built organizational routines. In fact, Microsoft strategy is from time to time said to evolve towards the software rental with continuous bug correction, improvements being also included in the rental price, which might indeed sound like Microsoft's answer to the Libre software threat<sup>20</sup>. To test this, we simply turn to a more dynamic version of our simulation exercises i.e. we now analyze diffusion dynamics when  $\beta$  is significantly increased during the diffusion process as soon as the number of Linux users reaches a given threshold defined as a percentage of the entire potential adopter populations<sup>21</sup>. Figure 2 presents results with again  $a = 2$  and  $b = 1$  while plotted variables  $\alpha$  and  $\beta$  correspond to their initial values. As the percentage of Linux adopters reaches 5%,  $\beta$  is increased by 0.2: Microsoft's reaction is therefore associated with a very low threshold, while the reaction itself is strong. Still, Linux diffuses in a way very similar to what happens without any such reaction or evolution (compare with figure 1). More precisely, when  $\beta$  is increased by 0.2 as the

---

classical but yet mainly empirical distinction between major and minor innovations: major innovations are innovations which are sufficiently better in order to diffuse, whereas minor innovations below this threshold will therefore never diffuse. As a consequence, minor innovations will most often to be rendered compatible with existing standards as they would otherwise have almost no chance of diffusion: the existence of diffusion thresholds endogenously creates incentives to compatibility for minor innovations. As a consequence, the existing standard gets continuously ameliorated instead of getting challenged by poor alternative candidates, which also appears an obviously socially optimal process.

<sup>20</sup> An example of which might be Microsoft's so-called ".NET" strategy.

<sup>21</sup> Another simulation methodology would have been to consider a continuous evolution of  $\beta$ . We prefer the one we have selected here because we consider that evolutions in business models are to be rare, as all producers - be they the Linux community - are organizations subject to routines.

percentage of Linux adopters reaches 5%, diffusion is roughly similar to what would happen with an initial value:  $\beta = \beta + \delta$  with  $\delta < 0.1$ .

Yet similar results<sup>22</sup> obtain when  $\alpha$  is increased during the diffusion process: results still hold when Linux users partly cease to be zealots after a while. We indeed interpret both results as a classical path-dependence effect (David, 1985): due to non-linear dynamics – a very strong attractor associated with a meta-stable state in trajectory space –, results appear as weakly sensitive to possible variations of state parameters during diffusion processes. Once diffusion has started, it is difficult to stop: to put it differently, (small) historical events do matter a lot, and proselytism induced by some sort of Microsoft-phobia will have played a similar role for Linux as early technological constraints have played for the QWERTY keyboard: proselytism might disappear soon, but it will have permanently oriented the diffusion path. Idiosyncratic historical conditions have given sufficient initial momentum to Linux diffusion: if local externalities had not been strong enough then the system would never have been able to “re-start”.

To summarize our results, tentatively<sup>23</sup> generalizing them to Libre software, Libre software is likely to benefit from very high market performances, even against installed proprietary standards, and even when their producers react quickly and strongly, because of a more efficient improvement rate but only as long as it is also sustained by strong local externalities at least at the beginning of its diffusion. To return now to our main ‘sustainability’ question, Libre software therefore appears as mainly sustainable in this respect, too, as long as it is able to deal with a critical *initial momentum* issue.

#### **4. Conclusion**

This last issue will of course be critical if we are some day to consider Libre software the same way as generic medicine, i.e. if we are to consider that there is a need for “generic” software: non-proprietary alternatives, for instance in the case of operating systems, middleware, and perhaps also in the case of a few key applications, i.e. generally for software which is so widely used and so subject to strong compatibility issues that there might be a need for a public intervention to correct market failures due to network effects. In these cases, supporting Libre software indeed appears as an excellent candidate for such an intervention: all the more so as Libre software still appears to some analysts as better for fixing bugs and re-creating existing programs rather than for creating new ones, as the necessary degree of co-ordination involved is in this latter case obviously higher. Then a major question for public agencies, and an agenda for future research, would be both to determine more precisely which software would be

---

<sup>22</sup> Available on demand from the authors.

<sup>23</sup> At least two major disclaimers apply, however. First, it would be necessary to analyze how these results vary for types of Libre software other than operating systems, considering different values of  $a$  and  $b$ , although they notably seem to hold with lower  $a$ . Further studies would also be needed to deal with more complex and realistic strategies from software producers. We have for instance considered improvements as continuously following adoptions, whereas strategies often rely upon the periodic release of non perfectly compatible versions. Since the situation presented is a marginal case, we conjecture that they will remain similar when timing between releases is small. When it is higher, with newer versions are not perfectly compatible with previous ones, we also conjecture that results would be even more favorable to Linux.

suitable for such issues<sup>24</sup>, and also what kind of actions can be undertaken to help launch Libre software development projects, i.e. both to create Libre communities and initial momentum in Libre software diffusion without relying on cultural feelings. Although it might still seem somewhat paradoxical, we have argued here that it might prove easier to build self-supporting development communities with their dedicated ancillary companies than to create sufficient initial momentum.

Then Libre software, born as a fad, would turn into an economic institution. In this respect, let us simply recall that both the scientific institution with its associated convention of « Open Science » and the intellectual property institution have also been born in rather strange circumstances: as a matter of fact, « Lettres de Patente » were once used to attract foreign inventors so that they would import technologies which already existed elsewhere (David, 1993) whereas scientists were originally “devoted” to raising princes’ reputation, which implied that they render their discoveries public (David, 1995). As a consequence, and specially in a knowledge-based (David & Foray, 1995) or idea-based (Romer, 1993) economic context, we should certainly start considering Libre software differently, against today’s still prevailing skepticism, since it is giving birth not only to an original incentive structure within Libre communities but also to a new and general economic model for the software industry, a model whose consequences have yet largely to be understood.

### ***Bibliography***

- Arthur W.B. (1989), Competing technologies, increasing returns and lock-in by historical events, *Economic Journal* 99: 116-131.
- Cohen M. D. (1983), Conflict and Complexity: Goal Diversity and Organization Search Effectiveness, *American Political Science Review*, 78 : 435-451.
- Codendet P., Créplet F., Dupouët O. (2000), Communities of practice and epistemic communities : a renewed approach of organizational learning within the firm, paper presented to WEHIA 2000 Conference, Marseille, June.
- Dalle J.-M. (1995), Dynamiques d’adoption, coordination et diversité, *Revue Economique*, 46: 1081-1098.
- Dalle J.-M. (1997), Heterogeneity vs. externalities: a tale of possible technological landscapes, *Journal of Evolutionary Economics* 7: 395-413.
- Dalle J.-M. (1998a), Heterogeneity and rationality in stochastic interaction models, in Cohendet P., Stahn H. (eds), *The economics of networks: behaviors and interactions*, Springer Verlag: Berlin, pp. 123-145.
- Dalle J.-M. (1998b), Local interaction structures, heterogeneity, and the diffusion of technological innovations, in Orléan A. & Lesourne J. (ed), *Self-organization and evolutionary approaches: new developments*, *Economica*: Paris, pp. 240-261.
- Dasgupta P., David P.A. (1994), Towards a new economics of science, *Research Policy* 23: 487-521.
- David P.A. (1985), Clio and the economics of QWERTY, *American Economic Review (Papers and Proceedings)* 75: 332-337.

---

<sup>24</sup> It would otherwise provide inefficient incentives for many proprietary software producers to adopt Libre models sponsored by public funding.

- David P.A. (1988), Putting the past into the future of economics, Technical Report 533, Institute for Mathematical Studies in the Social Sciences: Stanford University.
- David P.A. (1993), Intellectual property institutions and the Panda's thumb, CEPR publication n°287, Stanford University.
- David P.A. (1995), Reputation and agency in the historical emergence of the institutions of 'Open Science', paper presented to the National Academy of Sciences Colloquium on the Economics of Science and Technology, held at the Beckman Center, UC Irvine, 20-21 October 1995.
- David P.A., Foray D. (1995), Accessing and expanding the science and technology knowledge base, *STI Review* 16 : 13-68.
- David P.A., Foray D., Dalle J.-M. (1998), Marshallian externalities and the emergence and spatial stability of technological enclaves, *Economics of Innovation and New Technology* 6: 147-182.
- Durlauf S.N. (1993), Non-ergodic economic growth, *Review of Economic Studies* 60 : 349-366.
- Foray D. (1997), The dynamic implications of increasing returns: technological change and path-dependent inefficiency, *International Journal of Industrial Organization* 15: 733-752.
- Harhoff D., Henkel J., Von Hippel E. (2000), profiting for voluntary information spillovers: how users benefit by freely revealing their innovations, mimeo, 26p.
- Jullien N. (1999), Linux, la convergence du monde Unix et du monde PC?, Terminal.
- Kirman A.P. (1993), Ants, rationality and recruitment, *Quarterly Journal of Economics* 111: 137-156.
- Kirman A.P. (1998), The Economy as an Interactive System, in Arthur W.B., Durlauf S.N. & Lane D (eds), *The Economy as an Evolving Complex System II*, Addison Wesley : New York, pp. 491-531.
- Lakhani K., Von Hippel E. (2000), How Open Source software works : 'free' user-to-user assistance, MIT Sloan School of Management WP #4117.
- Lerner J., Tirole J. (2000), The simple economics of Open Source, mimeo, 38p.
- Ousterhout J. (1999), Libre software needs profit, *Communications of the ACM*, 42 : 44-45.
- Romer P. (1993), The economics of new ideas and new goods, Proceedings of the World Bank Annual Conference on Development Economics 1992, World Bank, Washington DC.
- Von Hippel E. (1988), *The sources of innovations*, MIT Press.