Proprietary or Open Source Software?
Winner-Take-All Competition, Partial Adoption and Efficiency

Eric DARMON * Thomas LE TEXIER *
Dominique TORRE *

December 2010

* CREM – Centre de Recherche en Economie et en Management
Université de Rennes 1
eric.darmon@univ-rennes1.fr
thomas.letexier@univ-rennes1.fr

* GREDEG - CNRS - University de Nice Sophia-Antipolis
250, bd. Albert Einstein, F-06560 Valbonne Cedex
dominique.torre@gredeg.cnrs.fr

Abstract

In this paper, we study the conditions ruling the diffusion of an Open Source software as opposed to a proprietary one. The two software differ according to their usage costs (adoption costs, existence of customized functionalities). We distinguish two categories of users, namely "end-users" and "users-developers", according to their ability to contribute to the development of the OS software. We characterize the Nash equilibria in a sequential game where players are first the producer of the proprietary software which chooses a price and a quality level, and second users who chose between adopting the OS, the proprietary software or not adopting. Due to the externalities of use and development, the final level of adoption of the proprietary software and of the OS solution depend on initial conditions relative to users expectations. Starting from identical initial conditions (adoption costs, magnitude of the externalities generated by the community of developers and users, etc.), a winner-takes-all competition may arise between the two software, which may lead to the crowding out of one of them. Other cases exhibit separating equilibria where users are distributed between the two software according to their aptitude to develop or to adopt. Such multiple equilibria being only imperfectly controlled by the commercial firm, we show that the strategy of the commercial firm can be understood as a balance between a low price – high quality strategy and a high price – low quality strategy. We analyse the qualitative properties of pooling and separating equilibria. The sole existence of a credible OS alternative improves the utility of end-users, even if only the proprietary solution is finally adopted. The diffusion of the OS software may sometimes generate conflicts of interests. In some cases, the division is not only between the users and the firm but the interests of some users can also be aligned to that of the firm.

Key words: Open Source – Proprietary Software – Increasing returns – Community of developers
JEL Codes: D3-D4-L1
Introduction

Competition outcomes that result from the introduction of ‘libre’ activities in the software market are analysed for several years. In particular, many economic and business models of hybridisation have been introduced by the software industry and discussed by academics. They analyse the way software firms can leverage from open source development projects for proprietary purposes (Hertel et al., 2003; Henkel, 2004; Dahlander and Magnusson, 2005; von Krogh and von Hippel, 2006; Harhoff and Mayrhofer, 2008). Within this framework, cooperative participative patterns have been shown to be beneficial for both community-based and commercial organizations provided that suitable governance mechanisms are defined. Although numerous examples of commercial-communautary alignments have been presented in the ‘libre’-related literature (Bonaccorsi et al., 2006; Dahlander and Wallin, 2006; Dahlander, 2007; Dahlander and Magnusson, 2008; Ghosh et al., 2008), few attempts have been at the same time carried out to link such a trend to the potential commercial outcomes a direct competitive face-off would provide, as well as the social benefits both producers and adopters may derive from it.

Previous competition model-based results have shown that there exist various ways of analyzing competition in the software market. They reveal that the co-existence of both types of software solutions is likely to depend on the environment in which software is produced (i.e. externalities), as well as on quality, functionalities, cost of proprietary solutions, and size of open source communities). Some authors find that open source software are likely to overcome proprietary ones from a competition viewpoint, for both informational (Kuan, 2002) and quality reasons (Bessen, 2002). In contrast, other papers show that firms have better chances to persist on markets than open source projects do through competitive processes (Casadesus-Masanell and Ghemawat, 2006).

Apart from such two extreme cases, most of the contributions reveal that both types of software production activities are likely to co-exist. Bonaccorsi and Rossi (2003) find for instance in their agent-based model proprietary and ‘libre’ software may co-exist when network effects are moderate. Besides, the model of Johnson (2003) stresses that the success of open source solutions over proprietary ones only applies when users exhibit a sufficient level of effort to contribute open source code. In a similar way, in a mixed duopoly model in
which producers differ in their objective function, Economides and Katsamakas (2006) found that competition outcomes may lead to a shared market situation by analyzing a two-sided competition model between an open source platform and a proprietary platform. Dalle and Jullien (2000; 2003) present a dynamical simulation model of diffusion of Linux in which both local and global network effects are considered. They point out the crucial role of early adopters on the success or the failure of open source projects and underline the major impact of compatibility and governance models on competition outcomes. As already evidenced in previous contributions, the co-existence of both types of software in the market is proved to be a possible competition issue.

Some theoretical works find competition outcomes unclear and dependant on the way commercial firms react to the ‘libre’ activity. Focusing on the impact of product-based heterogeneity, Bitzer (2004) obtains that incumbents (i.e., proprietary software firms) can remain profitable by setting up a higher price. Following the job signaling hypothesis formulated by Lerner and Tirole (2002), the model of Mustonen (2003) focuses on the role of wages paid by proprietary firms on competition issues. He shows that the commercial firm can crowd out the diffusion of the open source software solution by proposing higher wages and that both types of software may co-exist in the market. The duopoly model of Lanzi (2009) reveals that competition outcomes depend on the level of complexity of the open source software solution, inasmuch open source dominance may emergence if learning costs are low whereas a shared-market outcome appears when learning costs are higher-leveled. In a similar vein, Välimäki and Oksanen (2005) focus on the role of switching costs on software adoption patterns and highlight various competition outcomes. Introducing different types of innovation patterns and interoperability features, the dynamical model of Leoncini et al. (2008) evidences the perfect interoperability innovation pattern is likely to lead to a shared-market outcome whereas more ambiguous results are obtained when analyzing the other innovation patterns.

In our paper, we consider the adoption issue by explicitly taking into account the coexistence of heterogeneous users (i.e. users with different programming abilities and differentiated preferences). We use the typology of Franke and von Hippel (2003), by distinguishing developers-users and end-users. We define a theoretical framework in which motives to develop are due to the lack of needed specific functionalities provided by a proprietary software. Such a hypothesis attracts our attention since open development seems to be
preferred to proprietary software adoption in specific scientific fields like geophysics or radiology (Diviacco, 2005; Achleitner et al., 2006). Besides, our framework takes into account the difference in license costs when a user can substitute proprietary software to open source software released under GPL license terms. We then define a competition game theoretical model between a monopolistic firm producing proprietary software and a community developing alternative GPL-based open source software. As a twin issue, we propose to evaluate the welfare generated by the introduction of ‘libre’ solutions so as to measure the social effect of open source software that few studies have focused on so far (Schmidt and Schnitzer, 2003).

We characterize the Nash equilibria in a sequential game where the producer of proprietary software plays first and potential users play second. We show that the final level of adoption of both open source software and proprietary software depends on initial conditions relative to users expectations. Under some settings, a winner-takes-all competition may arise between the two types of software, which may lead to the crowding out of one of them. Other cases exhibit separating equilibria where users are distributed between the two types of software according to their ability to develop or to adopt and to their adoption costs. Such multiple equilibria are imperfectly controlled by the proprietary firm. The strategy of the commercial firm can thus be understood as a balance between a low price – high quality strategy and a high price – low quality strategy. By analyzing the qualitative properties of pooling and separating equilibria, we evidence that the sole existence of a credible open source alternative improves the utility of end-users, even if only the proprietary solution is finally adopted. However, the open source software diffusion is shown to generate conflicts of interests under some circumstances. In some cases, the divide not only stands between the users and the firm but the interests of some users can also be aligned to that of the firm.

The paper is organized as follows: Section 2 presents the model. Section 3 analyzes the Nash equilibria it generates and discusses its qualitative properties. Section 4 introduces welfare-related issues. Section 5 concludes.
2. The model – General settings

The model analyzes the interactions between a firm producing proprietary software and two categories of users, namely end-users and developers-users, according to their ability to contribute to the development of open source software.

We define the model as a two-step game. First, the firm defines a price-based and quality-based strategy. Second, potential users decide whether to adopt or not by taking open source development dynamics into account.

2.1. End-users and developers

There are two categories of users, namely developers ($D$, in proportion $\mu$ of the total population) and end-users ($EU$, in proportion $1-\mu$ of the total population whose size is set to 1). End-users and developers differ according to two properties.

First, developers are able to develop new functionalities and customize open source software by adding new lines of code, while end-users are not. In other words, as the source code is open, a developer is able to develop new functionalities for his own benefit. Once these functionalities are developed, there is no cost in releasing them to the whole community of developers and in giving end-users free access to these functionalities$^1$. Second, end-users and developers do not face the same adoption costs. Developers are generally expert users (i.e., engineers, computer scientists) and are used to adopt new software. On the contrary, end-users face high initial adoption costs while choosing new software. Besides, it seems reasonable to assume that these costs are generally higher when adopting open source software than those incurred when adopting proprietary one. One reason is that open source software are primarily developed and open source projects are led by expert users who cannot devote much time to make extensive tutorials/FAQs, numerous user-friendly interfaces, etc$^2$.

For all these reasons, we suppose that developers do not face any adoption costs while end-

---

$^1$ We restrict our analysis to GPL (General Public License) licenses. Other types of licenses (e.g., the BSD license) allow any user or developer to commercialize her own software derived from initial software. Such another type of license raises an additional incentive problem for developers who may be motivated to ‘close’ the code of their customized software instead of making it freely available to other users. This specific case could be studied as an extension of this article.

$^2$ It could be argued that users may incur an adoption cost even when adopting proprietary software. However, as user-friendliness is an important adoption factor, commercial firms devote important marketing efforts to maximize the overlook of their software. As such, it appears relevant to distinguish OSS from proprietary software by overlooking the adoption costs required when using proprietary software.
users do. Even inside the population of end-users, some agents are more likely to quickly adopt open source software. This varying ability translates into an heterogeneous adoption costs, which is here assumed to be uniformly distributed between 0 and $\epsilon^{EU}$.

2.2. Proprietary software and open source software

The two types of software need to perform two types of functionalities, i.e., generic and specialized functionalities. To be suitable for adoption, software needs to include all generic functionalities. Users (i.e., end-users and developers) are heterogeneous in their expectations for specialized functionalities. We suppose that they are uniformly distributed on a Salop-like circle of circumference 1. Agents are uniformly distributed on the circle and each agent needs the functionality related to her location. Users may choose (i) to adopt proprietary – closed source – software; (ii) to adopt GPL-based open source software; or (iii) not to adopt. In the last situation, we can suppose that users can manage to do the same task either manually or with a combination of previously acquired software (i.e., at no purchase cost). By convention, this default option leads to a null payoff.

2.3. Strategy of the proprietary firm

We suppose that a single commercial firm produces proprietary software. Because software are digital goods, its reproduction (or variable) costs are negligible (Shapiro and Varian, 1998): we assume them to be null and independent from the number of adopters. The development cost depends on the quality $q$ of the software that we assimilate to the extent of specialized functionalities covered by the software. By devoting more effort, the firm can develop more specialized functionalities in order to cover more specific needs on the unit circle. The quality $q$ is then expressed as a fraction of the unit circle circumference ($q \in [0,1]$) and its cost $g(q)$ is controlled by the firm. Wider are the existing functionalities, more it is difficult to widening them more without decreasing the adaptability of the software to each specific use (it is very costly to combine efficiently the functionalities of an adapted presentation program and of a sophisticated program of scientific calculus in the same software). We then suppose that the development cost $g(q)$ increases at an increasing rate on the quality and we specify it as $g(q) = \beta q^2,(\beta > 0)$.

Let us denote by $c_L (c_L > 0)$, the license fee paid by each user to the firm. In the case where
there is an univocal correspondence between the pair \((q, c_L)\) chosen by the firm and the proportion of proprietary software users, the profit of the firm producing the proprietary software is defined as (1):

\[
\max_{c_L \geq 0, q \in [0,1]} \pi = \sup \left( 0, n^p c_L - \beta q^2 \right)
\]

where \(n^p \in [0,1]\) denotes the proportion of users of proprietary software.

### 2.4. Utility generated by proprietary software adoption

The ability of both proprietary software and open source software to perform generic functionalities strongly depends on the total number of adopters. Such a network externality is motivated by several factors, such as the compatibility between users and the availability of complementary services and software. We therefore suppose that the utility associated to such functionalities is closely dependent on the number of users of the software solution considered. Consequently, we assume that the utility generated by the general needs \(f_1(n^p)\) positively depends on the adoption level expressed by \(n^p\) (with \(f_1'(.) > 0\), \(f_1(0) = 0\) and \(f_1(1) = k_i, (k_i > 0)\)). For convenience purposes, we set \(f_1(n^p) = \alpha n^p\).

As the source code of proprietary software is closed, no user can develop its own specialized functionalities. The utility generated by proprietary software then depends on whether the proprietary firm has developed the desired specific functionality or not. Users whose functionalities are covered by proprietary software get an additional utility equal to \(d\) (with \(0 < d\)), 0 otherwise. As software are experience goods, a user cannot fully anticipate at adoption time the specialized functionality she will need in the future, and has therefore to assess its ex post utility. Consequently, she can ex ante only consider the extent of specificity covered by proprietary software and try to anticipate the expected future utility of proprietary software. By convenience, we also suppose that users are risk-neutral. As a consequence, the expected utility of potential users relatively to the specific uses of proprietary software is set to be the weighted average of their respective ex post utilities according to their location (i.e., inside or outside the set of the functionalities effectively covered by proprietary software). Let \(d\) correspond to the level of utility generated by the specific uses of proprietary software. The expected utility derived from the specialized functionalities is then equal to \(dq\).
Once the license cost \( c_L \) charged by the firm has been deduced, the net utility derived from the adoption of proprietary software can be expressed as follows:

\[
u_i^p = \alpha_i n^p + dq - c_L\]

(2)

**Figure 1: The utility of proprietary software**

2.5. The utility generated by open source software adoption

When adopting open source software, users do not have to pay a license fee. As assumed in the case of proprietary software adoption, we suppose that open source software needs to meet all the generic functionalities to be suitable for adoption\(^3\). As previously mentioned, generic functionalities are subject to a direct externality. Besides, developers may – in the case of open source software – freely modify and customize the initial software solution. The modular architecturing of most recent open source software stimulates such customization features so that open source developers may add functionalities beneficial to all the users. As open source software is maintained and managed by developers, the utility derived from specialized functionalities is highly dependent from the number of contributors \( n_D^{OS} \). In other terms, the larger the number of developers who contribute to the code is, the more functionalities are added, and the higher the utility of open source software is for all the users (i.e., developers and end-users). This positive link is introduced by adding a positive

\(^3\) We may suppose that these functionalities are ex ante developed by ‘kernel’ developers and/or by the project manager so that OSS exists once all these functionalities are fully available (e.g., “1.0” software version release).
externality \( f_2(n_{D}^{OS}) \) on both users and developers utilities \( (f_2'(k_2) > 0, f_2(0) = 0, f_2(1) = k_2) \). By convenience, we set \( f_2(n_{D}^{OS}) = \frac{\alpha_2 d}{\mu} n_{D}^{OS} \) with \( \alpha_2 > 0 \).

The adoption cost \( c_i^{EU} \) \( (0 \leq c_i^{EU} \leq \bar{c}^{EU}) \) of any user \( i \in [0,1] \) is deduced from its gross utility. Developers and end-users are classified according to the level of their adoption costs. Consequently, user \( i \) is a developer and her adoption cost is null when \( i \in [0, \mu] \). End-users are ranked on the segment \( i \in [\mu,1] \) according their adoption costs that increase at a linear rate on the segment. The adoption cost of the end-user \( i \) is by definition \( c_i^{EU} = \left(\frac{i-\mu}{1-\mu}\right)\bar{c}^{EU} \) (cf. figure 2).

![Figure 2: Open source software-related adoption costs](image)

We then express the utility associated to open source adoption by agent \( i \) \( (i \in [0,1]) \) as follows:

\[
u_i^{OS} = \alpha_i n^{OS} + \frac{\alpha_i d}{\mu} n_{D}^{OS} - c_i^A \quad \text{with} \quad c_i^A = \begin{cases} 0 \text{ when } i \in [0, \mu] \\ c_i^{EU} \text{ when } i \in [\mu,1] \end{cases}
\] (3)
3. The sequential game

The firm knows the different characteristics of the potential users \((\mu,\alpha_1,\alpha_2,...)\) of the software. It makes its choices initially and users reply subsequently to the firm’s strategy by formulating their own adoption choice. The firm must then anticipate the answers of developers and end-users to its price and quality decisions. It has however then a complete but imperfect information about the possible strategies of the potential adopters of its proprietary software. It knows the different characteristics of the potential users \((\mu,\alpha_1,\alpha_2,...)\) of the software solution but it can imperfectly anticipate their choices when many possibilities are equivalent. Since we can suppose that the price and the quality of software cannot be frequently altered, we assume that firms’ and users’ choices are sequential. The structure of the sequential game is depicted by figure 3:

---

**Step 1**: The commercial firm sets the license cost \((c_L)\) and the quality \((q)\) of its software

**Step 2**: software adoption of user \(i\)

---

Figure 3: Structure of the sequential game (OS = Open Source Software adoption; P = Proprietary Software adoption; \(\emptyset\) = No adoption)

At the first step of the game, the firm chooses the extent of functionalities \(q^*\) and the level of its license fees \(c_L^*\) which maximize its expected profit. At the second step of the game, by taking such levels into account, each user \(i\) compares its respective utilities as proprietary software user (i.e., ‘P’ strategy), open source software adopter (i.e., ‘OS’ strategy) or non
adopter (i.e., ‘∅’ strategy). The individual utilities of users depend on each user’s expected level of adoption of each software. We suppose that these expectations – be it right or wrong – are the same for all users. An eductive process generates a convergence of wrong expectations towards the locally stable equilibrium of the area of stability to which they belong.

We solve the game by backward induction. Indeed, we first consider the second sub-game, that is to say the adoption decisions of developers and end-users for a given strategy of the firm. We then consider the first sub-game, by determining the profit-maximizing strategy of the firm.

3.1. Second step of the game

Let us thus consider a couple \((q,c_L)\). Once we note the expectations \(\bar{n}^P, \bar{n}^{OS}, \bar{n}_D^{OS}\) relative to the respective sizes of proprietary software customers population, the total population of OS end-users and the population of OS developers, the choice of any user \((i, i \in [0,1])\) is as follows:

- Choosing \(P\) if \(\alpha_i \bar{n}^P + dq - c_L \geq \sup \left\{ 0, \alpha_i \bar{n}^{OS} + \left( \frac{\alpha_i d}{\mu} \right) \bar{n}_D^{OS} - c_i^A \right\}\)
- Choosing \(OS\) if \(\alpha_i \bar{n}^{OS} + \left( \frac{\alpha_i d}{\mu} \right) \bar{n}_D^{OS} - c_i^A \geq \sup \left\{ 0, \alpha_i \bar{n}^P + dq - c_L \right\}\)
- Choosing \(∅\) if \(0 \geq \sup \left\{ \alpha_i \bar{n}^P + dq - c_L, \alpha_i \bar{n}^{OS} + \left( \frac{\alpha_i d}{\mu} \right) \bar{n}_D^{OS} - c_i^A \right\}\)

Each agent associates her own optimal choices to each expected level of \(\bar{n}^P, \bar{n}^{OS}\) and \(\bar{n}_D^{OS}\). Let us conventionally assume that, if they expect the level \(\bar{n}^{OS}\), they will consistently suppose that this population only gather developers if \(\bar{n}^{OS} \leq \mu\). If \(\bar{n}^{OS} > \mu\), this population also integrates the end-users whose adoption costs are the lowest\(^4\). From the aggregation of these optimal decisions, the effective proportions of proprietary software users, of developers and end-users of OSS (i.e., \(n^P, n^{OS}\) and \(n_D^{OS}\) respectively) are then deduced by the expected levels

\(^4\) This kind of assumption is frequently used in adoption models (see for instance Crémer, 2000).
of each populations (see appendix 1). Hence, for each couple \((q, c_L)\), the condition 
\((n^P, n^{OS}, n^{D}) = (\tilde{n}^P, \tilde{n}^{OS}, \tilde{n}^{D})\) defines the Nash equilibria of the sub-game corresponding to
the second step of the game.

**Proposition 1.** Whatever the pair \((q, c_L)\) selected by the firm at the first step of the game, there exists at least one Nash equilibrium associated with Step 2 sub-game.

**Proof of proposition 1.** See appendix 1.

Without any specific restriction on the values of the parameters, the competition between OS and proprietary software could provide tautological outcomes. We then introduce assumption 0 that excludes the lack of viability of the proprietary solution when there is no open source.

**Assumption 0.** The activity of the proprietary software firm is always profitable when the open source development activity is not introduced.

Assumption 0 states that there exists an economically viable pair \((q^*, c^*_L)\), i.e., a quality/price solution for the proprietary software firm, such that i) there exists an expected size level \(\tilde{n}^{P*}\) of proprietary software users high enough to incitate users to adopt the proprietary solution when there is no alternative, and ii) for this pair \((q^*, c^*_L)\), the profit of the firm 
\(\pi = c^*_L - \beta (q^*)^2\) is non-negative. Assumption 0 allows formulating Proposition 2:

**Proposition 2.** Under Assumption 0, the distribution of users \(\{n^{P*} = 1, n^{OS*} = 0, n^{D*} = 0\}\) corresponding to the full adoption of proprietary software is always a Nash equilibrium of the second-step sub-game.

**Proof of proposition 2.** See appendix 1.

Proposition 2 establishes that the viability of the proprietary software alone is a sufficient condition to obtain a Nash equilibrium where the OS software is crowded out (or its partial adoption not possible).
With assumption 0, the second step sub-game has at least one equilibrium. At this equilibrium, the open source development activity is crowded out and all users adopt proprietary software. But other equilibria may exist. Consider for instance the (limit) case where the adoption cost $c_i^A$ of the OS software vanishes whatever $i$. With a value of $q$ sufficiently low enough and a value of $c_L$ sufficiently high, the inequality $f_1(0) + dq - c_L \leq 0 \leq f_1(1) + dq - c_L$ may hold. Since in this case $f_1(1) + f_2(\mu) - c_i^A$ is non-negative whatever $i$, the condition $\sup \{0, f_1(0) + dq - c_L\} \leq f_1(1) + f_2(\mu) - c_i^A$ also applies for all the users. The outcome $\{n^P = 0, n^D_{OS} = \mu, n^{OS} = 1\}$ is then a second equilibrium of the second set sub-game. By continuity, the same result is maintained when $c_i^A = \varepsilon_i (\forall i)$ with $\varepsilon_i$ close to 0. Thus, the equilibrium $\{n^P = 0, n^D_{OS} = \mu, n^{OS} = 1\}$ exists for a non-empty range of variation of $c_i^A (\forall i)$. However, as $c_i^A$ is far from being null for a part of end-users population, this “all OS” adopters equilibrium does not exist for all possible values of $\varepsilon^{EU}$. Last, for some values of $\varepsilon^{EU}$, other equilibria may exist. Unlike the “pooling” equilibria $\{n^* = 1, n^D_{OS} = 0, n^{OS} = 0\}$ and $\{n^* = 0, n^D_{OS} = \mu, n^{OS} = 1\}$, those equilibria are “separating” ones, that is to say they split users into two sub-populations, the proprietary software users and the OS users. When the functions $f_1(\cdot), f_2(\cdot)$ are not linear and and $g(\cdot)$ is not quadratic, there may even be many separating equilibria. Though our linear - quadratic specification limits the possibilities, the multiplicity of the second step sub-game Nash separating equilibria is a possible outcome of the second step sub-game.

We then introduce a second reasonable assumption parallel to assumption 0.

**Assumption 1.** The higher level of adoption costs that users may face is upperly bounded so that $\varepsilon^{EU} \leq (\alpha_i + d\alpha_z)$.

By this assumption, we suppose that the end-user exhibiting the largest adoption cost $\varepsilon^{EU}$ always gets a non-negative utility when it adopts OSS. This assumption is somewhat comparable to the conditions a proprietary software firm has to face for its activity to remain profitable.

We add a third assumption making consistent the agents expectations.
**Assumption 2.** Agent $i$ cannot simultaneously adopt open source software and expect that a proportion of agents less than $i$ adopt it.

We exclude with assumption 2 the total inconsistency between expectations and choices. We suppose that agents are aware of their rank among other users and are likely to integrate this information in their adoption decisions by excluding some expectations inconsistent with this hierarchy.

We then can exhaustively analyze the nature and stability of the second step sub-game equilibria of the linear-quadratic specification of the model (see appendix 1). At such a stage, the firm has already fixed $c_L$ and $q$ at the first stage of the game. The remaining variables – which have to be determined by the users – are $n^P*$ (i.e., the Nash equilibrium proportion of agents adopting proprietary software), $n^{OS}* $ (i.e., the Nash equilibrium proportion of agents adopting open source software) and $n_{OS}^{D*}$ (i.e., the Nash equilibrium proportion of developers among agents adopting open source software). According to our specifications of $f_1(\cdot), f_2(\cdot)$ and $g(\cdot)$, and given assumptions 0,1 and 2, the typology of Nash equilibria of this second step sub-game can be exhaustively analyzed by considering the possible values of parameters $\alpha_1$, $\alpha_2$, $c^{EU}$, $\mu$ and $d$, as well as the step 1 variables of control $q$ and $c_L$. These cases are presented graphically in appendix 1 and summarized in table 1.

<table>
<thead>
<tr>
<th>NUMBER OF EQUILIBRIA</th>
<th>NUMBER OF STABLE EQUILIBRIA</th>
<th>TYPE OF STABLE EQUILIBRIA</th>
<th>CONDITIONS ON PARAMETERS AND ON STEP 1 CONTROL VARIABLES</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Case 1</strong></td>
<td>3</td>
<td>2</td>
<td>proprietary software only [P]</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$\alpha_1(1-\mu) &gt; c^{EU}$, $dq - c_i &lt; \alpha_2(2\mu - 1) + d\alpha_i$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>OS software only [OS]</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$\alpha_1(1-\mu) &lt; c^{EU}$, $dq - c_i &lt; \alpha_2(2\mu - 1) + d\alpha_i$</td>
</tr>
<tr>
<td><strong>Case 4</strong></td>
<td>3</td>
<td>2</td>
<td>proprietary software only [P]</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$\alpha_1(1-\mu) &gt; c^{EU}$, $dq - c_i &lt; \alpha_2(2\mu - 1) + d\alpha_i$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>OS software only [OS]</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$\alpha_1(1-\mu) &lt; c^{EU}$, $dq - c_i &lt; \alpha_2(2\mu - 1) + d\alpha_i$</td>
</tr>
</tbody>
</table>
| Case 2 | 3 | 2 | proprietary software only [P] | $\alpha (1 - \mu) > \overline{\alpha}$  
$\nu d q - c_i > \alpha (2 \mu - 1) + d \alpha_i$  
$\nu d q - c_i < \alpha_i + d \alpha_i - \overline{\alpha}$  
OS software only [OS] |
|-------|---|---|-----------------------------|--------------------------------|
| Case 5 | 3 | 2 | proprietary software only [P] | $\alpha (1 - \mu) < \overline{\alpha}$  
$\nu d q - c_i > \alpha (2 \mu - 1) + d \alpha_i$  
$\nu d q - c_i < \alpha_i + d \alpha_i - \overline{\alpha}$  
OS software only [OS] |
| Case 3 | 1 | 1 | proprietary software only [P] | $\alpha (1 - \mu) > \overline{\alpha}$  
$\nu d q - c_i > \alpha (2 \mu - 1) + d \alpha_i$  
$\nu d q - c_i < \alpha_i + d \alpha_i - \overline{\alpha}$ |
| Case 6 | 1 | 1 | proprietary software only [P] | $\alpha (1 - \mu) < \overline{\alpha}$  
$\nu d q - c_i > \alpha (2 \mu - 1) + d \alpha_i$  
$\nu d q - c_i < \alpha_i + d \alpha_i - \overline{\alpha}$ |
| Case 7 | 3 | 2 | proprietary software only [P] | $\alpha (1 - \mu) < \overline{\alpha}$,  
$\nu d q - c_i < \alpha (2 \mu - 1) + d \alpha_i$  
$\nu d q - c_i > \alpha_i + d \alpha_i - \overline{\alpha}$  
coexistence of the two software [OS-P] |

Table 1: Nash equilibria of the second step of the game

In some cases (i.e., cases 1, 4, 2 and 5), there exist three Nash Equilibria, only two of which are stable. In these cases, either proprietary software or OSS is adopted at equilibrium but both software never simultaneously co-exist at the end of eductive process. These cases exhibit a “winner-takes-all” situation. The area of attraction of each “pooling” equilibria (see appendix 1) is defined by the location of the unstable “separating” equilibrium (E3). When the initial agents’ expectations about the adoption of OSS are below a critical level, proprietary software is finally adopted by all the users at the end of the eductive process. On the contrary, when expectations are beyond this critical level, OSS is finally found to prevail. Other cases correspond to a single stable “pooling” equilibrium (i.e., cases 3 and 6) where either proprietary software or OSS is adopted. Case 7 corresponds to a situation in which there are two stable equilibria. The first equilibrium is a pooling one and represents an outcome in which proprietary software firm fully covers the market, whereas the second one is separating one for which both software producers share the market.
These preliminary results stand for one given \((q, c_L)\) strategy of the firm. Solving backward, we define the equilibria of the first sub-game that maximizes the profit of the proprietary software firm.

### 3.2. First step of the game

The first step of the game corresponds to the choice of the pair \((q, c_L)\) maximizing the expected profit of the firm, given the possible \textit{ex post} adoption scenarios.

At this stage, the firm has to choose the pair \((q, c_L)\) maximizing its profit function. As the second sub-game equilibrium is generally not unique, the firm has to integrate each possible stable outcome associated with the second stage of the game ensuing from the pair \((q, c_L)\). The occurrence of these stable equilibria and their corresponding payoffs are also to be considered. The behavior of the firm towards risk can influence the shape of its objective function. For simplification purposes, we suppose that the firm is risk-neutral and only considers the expected profit generated by the different outcomes of the second step sub-game as answers to its choices at the first step of the game\(^5\). The expected profit of the firm associated to a given pair of control variables \((q, c_L)\) uses an objective or subjective distribution of probability on the occurrence of the stable equilibria of the second step sub-game. The form of this distribution has obviously an influence on the decision of the firm. As information of the firm is imperfect, it reasonably anticipates a uniform distribution on \([0,1]\) of the level of adoption \(\hat{n}_{\text{OS}}\) of the OS software. We label respectively \(\{n_{\text{OS}}^+, n_{\text{OS}}^{++}\}\) and \(\{n_{\text{OS}}^{++}, n_{\text{OS}}^{+++}\}\) the populations corresponding respectively to an instable and stable second step Nash equilibrium. Given assumption 2, the size of the respective stability areas of the second step sub-game equilibria then provides the required distribution of probability generating the expected profit of the firm that expressed as (4):

\[
\pi = \max_{c_L \geq 0, q \in [0,1]} \left[ 0, \left(p^* + p^{++} n^{+++} (c_L, q)\right) c_L - \beta q^2 \right] 
\]  

\(^5\) We make this assumption because the unstable equilibrium has a zero probability to occur.
where

- \( p^* = 1 \) and \( p^{\dagger} = 0 \) when \( (c_L, q) \) and \((\alpha_1, \alpha_2, d, \mu, \overline{c}^{EU})\) correspond to cases (3) and (6) of Table 1,
- \( p^* = n^{P^1} \) and \( p^{\dagger} = 0 \) when \( (c_L, q) \) and \((\alpha_1, \alpha_2, d, \mu, \overline{c}^{EU})\) correspond to cases (1), (2), (4) and (5) of Table 1,
- \( p^* = n^{P^1} \) and \( p^{\dagger} = (1 - n^{P^1}) \) when \( (c_L, q) \) and \((\alpha_1, \alpha_2, d, \mu, \overline{c}^{EU})\) correspond to case (7) of Table 1.

Given assumption 2, the size of the respective stability areas of the second step sub-game equilibria then provides the required distribution of probability generating the expected profit of the firm.

**Proposition 3.** An optimal outcome always exists for the firm at the first step of the game.

**Proof of proposition 3.** See appendix 2.

**Corollary.** An equilibrium of the two-step game always exists. When this equilibrium is not unique, equilibria correspond (except for cases of vanishing measure) to a single level of functionalities and license cost of proprietary software, but to different levels of adoption of proprietary software and OSS.

A comparison of cases (1), (2) and (3) illustrates the nature of the choice made by the firm at the first step of the game. In these 3 cases, the same condition \( \alpha_1(1 - \mu) > \overline{c}^{EU} \) is fulfilled on 3 parameters among 5. According the last two - namely \( d \) and \( \alpha_2 \) which characterize the efficiency of the effort of the firm and of the adoption externalities -, the firm decides to choose a relatively low level of quality and high level of price with a probability of success less than one in cases (1) and (2) or a relatively high level of quality and low level of price and the certainty to impose the software in case (3). In cases (1) and (2), the profit is then large in case of success but the success is not certain; in case (3), the profit is quite small but it is certain.
We next present three numerical examples to analyze possible competition outcomes.

3.2. Some illustrative examples

In our first example, we consider values of parameters corresponding to the cases 1, 2 or 3 \((\alpha_1 = 1, d = 1, \alpha_2 = 1, \beta = 0.2, \mu = 0.2, \sigma_{EU} = 0.1)\) according to which the direct externality outweighs the adoption cost incurred by the marginal user. In this case, let us bear in mind that the final outcome is always a winner-takes-all one, which leads to the crowding out of one of the two software producers. We plot the expected profit of the firm for different \((q, c_L)\) strategies\(^6\).

---

\(^6\) For each strategy, we consider the relative positions of the \(U^P\) and \(U^{OS}\) curves. From this comparison, we deduce the case (1-2-3) and compute the expected profit and the expected diffusion of the OS project. If \(U^P < 0\ (\forall t \in [0, 1])\), no user adopts proprietary software. Therefore, the firm does not make any investment on quality-based features, and the profit of the firm is null.
The maximum expected profit \( \pi^{\text{max}} \) is here strictly positive and reached for an interior value of the strategy set \( \left( c^*_L = 0.6 \text{ and } q^* = 0.2 \right) \). The right figure depicts the expected diffusion of OSS, i.e., the probability that OSS is diffused to the whole population. It shows how the proprietary firm can indirectly – and partially – control the diffusion of OSS by setting different price/quality strategies. When it chooses a “low price – high quality” strategy, the firm can crowd the OSS community out of the market (i.e., the expected OS diffusion is closed to 0). However, we see in this example that the firm has no incentive to do so, due to the development costs. Instead, the expected diffusion of OSS that corresponds to the firm’s optimal strategy is found to be equal to 0.91.

As such, the firm faces a trade-off, inasmuch as it may either select a strategy that crowds out the OS project by setting a high quality and/or low license cost, or it may apply a “low quality and/or high price” strategy. In the latter case, the firm gets a maximal profit when the “P” equilibrium is reached but it faces the risk to be crowded out (i.e., OSS to be massively adopted). The optimal behaviour of the firm has to be understood as a balance between these two strategies. We could also mitigate this result by taking into account the firm’s risk aversion. However, we suggest that such a change would not qualitatively change our results.
rather applying an optimal ‘conservative’ strategy (i.e., a quality increase and/or price decrease strategy).

A second example depicts a situation in which the needs of the users are more specific. Here, the add of specialized functionalities is of major importance for adoption purposes. In this example, we set \( d = 5 \) instead of \( d = 1 \).

![Figure 6: Expected profit of the firm (left) and OS expected diffusion (right) as a function of the quality and license cost charged by the firm.](image)

\[
(\alpha_1 = 1, d = 5, \alpha_2 = 1, \beta = 0.2, \mu = 0.2, \varepsilon^{EU} = 0.1); \quad \pi^{\text{max}} = 0.08 > 0 \text{ reached for } \begin{cases} c_L^* = 2.5 \text{ and } q^* = 1 \end{cases}.
\]

The profit-maximizing strategy aims here at developing a proprietary software solution which provides the largest range of specialized functionalities \( q^* = 1 \). Hence, the firm has an incentive to invest in quality in order to differentiate its software from OSS. We note that the expected diffusion of OSS is found to be equal to 0.88. As such, the optimal strategy of the firm is not to crowd OSS out of the market.

As opposed to the second example, the third example presents a situation in which the needs of the users are generic. Here, the presence of general functionalities shapes adoption outcomes. Numerical simulations are led by setting \( d = 0.1 \).
Figure 7: Expected profit of the firm (left) and OS expected diffusion (right) as a function of the quality and license cost charged by the firm

\( \left( \alpha_1 = 1, d = 0.1, \alpha_2 = 1, \beta = 0.2, \mu = 0.2, \bar{c}^{eu} = 0.1 \right) ; \ \pi^{\text{max}} = 0.124 > 0 \ \text{reached for} \ \left( c^*_L = 0.4 \ \text{and} \ q^* = 1 \right) . \)

Our results reveal that the firm does not here develop any specific functionality \((q = 0)\). Since users give little importance to specific functionalities, the optimal strategy of the firm is to focus on generic functionalities and therefore not to invest in specialized ones. As already evidenced in our two previous examples, we find that the firm has no interest in crowding out the OSS activity.

4. Market failures and the welfare properties of the introduction of OSS

The weight of the development externalities generated by the size of the community of OSS developers from one hand, and the role of the consumers’ externalities associated to both OSS and proprietary software use of proprietary and OS software then often generate multiple equilibria associated to the same decision of the proprietary software editor. The nature of the emerging equilibrium is in this case related to the form of expectations.

Grounding on the previous results, we can have two types of welfare analysis. A first issue is about multiple equilibria. We have seen that in many cases, a single firm strategy can lead to multiple equilibria. If so, is it possible to rank the possible outcomes. Second, these outcomes may be compared to a base case “prior” to the existence of OS projects, as the firm acted as monopolist. We still conducted both analyses in the case of a linear-quadratic specification.

When the optimal quality / price pair \((q^*, c^*_L)\) involves the occurrence of multiple equilibria, market failure emerges. Consider for instance in the linear-quadratic specification, the case
where \((q^*, c_L^*)\) generates two stable pooling equilibria, \(\{n^{P*} = 1, n^{OS*} = 0, n^{OS*} = 0\}\) and \(\{n^{OS**} = 0, n^{OS**} = \mu, n^{OS**} = 1\}\). The comparison of these pooling equilibria provides the following result in the linear-quadratic case:

**Proposition 4:** When the optimal price / quality pair \((q^*, c_L^*)\) is associated with two pooling stable equilibria at the second step sub-game,

i) if there is a coordination failure, the high level equilibrium is always the 'all-proprietary' one. All things equal, this case prevails when development externalities are relatively low.

ii) if there is a conflict of interest, the conflict rises between developers and part of or all end-users on the one hand and the firm, with - or without - some other end-users on the other hand. All things equal, this case corresponds to high development externalities.

Proof of proposition 4. See appendix 3

The first part of the proposition points out cases where the low level of expectations relative to the adoption of proprietary software may finally involve an "all OS" solution, which is not desirable even for developers. In the opposite situation, the second part of the proposition establishes that when externalities of development are rather high, the "all OS" solution is the more desirable for developers and part of end-users, while the other part of end-users and the firm would have preferred, if possible, the "all proprietary" equilibrium.

These results may be complemented by an equivalent proposition in the case where one of the stable solutions is a separating equilibrium:

**Proposition 5:** When the optimal price / quality pair \((q^*, c_L^*)\) is associated with one 'all-proprietary' and one separating equilibrium at the second step sub-game,

i) if there is coordination failure, the high level equilibrium is always the 'all-proprietary' one. All things equal, this case corresponds to low development externalities.
ii) if there is a conflict of interest, the conflict is between developers and part of all end-users on the one hand, and the firm and the other end-users on the other hand. All things equal, this case corresponds to high development externalities.

**Proof of proposition 5. See appendix 3**

Proposition 5 confirms proposition 4. Due to the existence of externalities in each software, the separating equilibrium is not efficient, even if the firm is not considered in the analysis of welfare. Interestingly, Propositions 4 and 5 show that an even larger diffusion of OSS is not always beneficial for users. Yet, these two propositions identify some cases where this diffusion can be detrimental to users surplus. Furthermore, Proposition 5 underlines that the incentives of some users and of the firm can be aligned in some situations. Propositions 4 and 5 however contrasts with the following result:

**Proposition 6:** Suppose that a (unique or not) pooling 'all proprietary' equilibrium is associated with the optimal price / quality pair $(q^*, c^*)$ in a setting in which agents can choose between proprietary software and the OS solution. Then this equilibrium is preferred by all users to the 'all proprietary' equilibrium existing under the same settings without any possibility to choose OSS. (proof: see Appendix 3)

Proposition 6 counterbalances propositions 4 and 5. Even if the OS solution is not actually selected and adopted, the existence of a 'potential competitor' for the proprietary solution provides incentives for the producer to lowering prices or improving quality of its proprietary software. The outcome is always positive for all users. Propositions 4 and 5 compare the properties of the two possible equilibria when OSS is already present. Together, they show that the magnitude of the development externality is crucial to determine the agents that benefit or are harmed when a particular equilibrium is selected. Interestingly, they show that, when a conflict of interest arises, in some cases, this conflict is not only between the firm and the whole population of users. Instead, it establishes that the interests of some users (those with high adoption cost) are associated to that of the firm.
5. Discussion and further research

This paper depicts the competition and coordination issues arising from the existence of OS projects: in a framework where externalities favour lock-in and winner-takes-all situations, the competition between OS and commercial software have been analyzed using a two-step game: in a first step, the firm sets the quality and the price of its software. In a second step, end-users and developers react by adopting the commercial software or engaging into the OS project. We have shown that such a situation often leads to multiple equilibria. Because the commercial firm can influence but cannot entirely select the most favourable equilibrium, it faces a trade-off: either it adopts a “low price – high quality” strategy and deters the OS project; or it adopts a “high price – low quality” strategy and bears the risk of being crowded out by the OS software. We show that the strategy of the commercial firm can be understood as a balance between these two basic strategies. Depending on the preferences of the users, we have pointed out some cases where i) the firm has no interest to invest in quality or ii) on the contrary, it has an incentive to develop all functionalities to maximize its profit.

These first findings can be discussed in several directions. First, we need to go further into the numerical exploration of different scenarii. One issue is how the efficiency of the organization of the OS project (captured by $\alpha_2$ in our model) impacts on its potential diffusion. This impact can be analyzed within our framework, by studying the reaction of the firm to alternative degrees of efficiency. Secondly, we could extend the current analysis by considering an additional step during which the firm could revise its price-quality strategy. This could be done using a repeated game. However, such extension would raise many additional problems since many choices are not easily reversible (adoption of users, quality of the firm).

Further, one could consider alternative strategies concerning the quality policy of the commercial firm. For example, Lerner and Tirole [2001] argue that, in letting their own developers implicated in OS projects, firms can understand competition better, by innovating as well as by using OS ideology to improve their production as detecting skillful developers. Dahlander and Magnusson [2005] stress that the firms may directly benefit from the OS communities by managing their activities, as the knowledge required to develop a software is not only controlled by the firm but also by the surrounding communities. As the addition of a specific functionality may sometimes be too costly, Krishnamurthy [2003] points out that
firms may be encouraged to support OS projects to lower their production costs. Such arguments leave room for a further analysis of new forms of cooperation between OS and proprietary software.

References

CREMER J. and GAUDEUL A. [(2004)], Some economics of the open-source software, Réseaux, (forthcoming),
DALLE J.-M. and JULLIEN N. [2003], 'Libre' Software: Turning Fads into Institutions?, Research Policy, 32:1, 1-11
GAUDEUL A. [2003b], Open Source Software Development Patterns and License Terms, GREMAQ WP.
LERNER J. and TIROLE J. [2001], The Open Source Movement: Key Research Questions, European Economic Review, 45:4-6, 819-826.
SHAPIRO C. and VARIAN H. R. [1999], Information rules: A strategic guide to the network economy, x, 352.
VALIMAKI M. and OKSANEN V. [2005], The impact of free and open source licensing on operating system software markets, Telematics and Informatics, 22:1-2, 97-110.
Appendix 1: Nash equilibria of the Step 2 sub-game

Proof of Proposition 1: Given that $n_{D}^{OS} = \inf \left[ n_{D}^{OS}, \mu \right]$, consider the choice made by agent $i$ when she observes the pair $(q_i, c_i)$ at the second step of the game:

- Choosing $P$ if $\alpha_i \bar{n}_i^P + dq - c_i \geq \sup \left\{ 0, \alpha_i \bar{n}_i^{OS} + \left( \frac{\alpha_i d}{\mu} \right) \bar{n}_i^{OS} - c_i^4 \right\}$
- Choosing $OS$ if $\alpha_i \bar{n}_i^{OS} + \left( \frac{\alpha_i d}{\mu} \right) \bar{n}_i^{OS} - c_i^4 \geq \sup \left\{ 0, \alpha_i \bar{n}_i^P + dq - c_i \right\}$
- Choosing $\emptyset$ if $0 \geq \sup \left\{ \alpha_i \bar{n}_i^P + dq - c_i, \alpha_i \bar{n}_i^{OS} + \left( \frac{\alpha_i d}{\mu} \right) \bar{n}_i^{OS} - c_i^4 \right\}$

The analysis of the second step sub-game consists in proving the existence of fixed points to the correspondence $T' = \left\{ t^P, t^{OS} \right\}$ from $\left\{ [0,1], [0,1], [0,1] \right\}$ in itself, where $T'$ associates - for all values of the parameters $(\alpha, \alpha_i, d, \mu, \tau^{OS})$ and of the optimisation variables $(c_i, q)$ - the actual population $(n^P, n^{OS}, n^O)$ of proprietary software users OSS users and non-adopters, to the set $(\bar{n}^P, \bar{n}^{OS}, \bar{n}^O)$ of the same expected populations. Since $\bar{n}^O$ and $n^O$ are defined respectively as $\bar{n}^O = 1 - \bar{n}^P - \bar{n}^{OS}$ and $n^O = 1 - n^P - n^{OS}$, the correspondence $T'$ has the same properties than the correspondence $T = \left\{ t^P, t^{OS} \right\}$ from $\left\{ [0,1], [0,1], [0,1] \right\}$ to itself which associates the pair of realised populations $(n^P, n^{OS})$ to the expected one $(\bar{n}^P, \bar{n}^{OS})$. As $T'$ is defined from a compact on itself, the continuity of $\alpha_i, \alpha_i$ and $\tau^{OS}$ is sufficient to prove the existence of at least a fixed point of $T'$ from which we then could deduce the existence of a fixed point of $T$. Let us define $T' = \left\{ t^P, t^{OS} \right\}$. Whatever $(\alpha, \alpha_i, d, \mu, \tau^{OS})$, $(c_i, q)$ and $(\bar{n}^P, \bar{n}^{OS})$, $n^P$ corresponds to the subset of $i \in [0,1]$ such that $\alpha_i \bar{n}_i^P + dq - c_i \geq \sup \left\{ 0, \alpha_i \bar{n}_i^{OS} + \left( \frac{\alpha_i d}{\mu} \right) \bar{n}_i^{OS} - c_i^4 \right\}$ while $n^{OS}$ corresponds to the subset of $i \in [0,1]$ such that $\alpha_i \bar{n}_i^{OS} + \left( \frac{\alpha_i d}{\mu} \right) \bar{n}_i^{OS} - c_i^4 \geq \sup \left\{ 0, \alpha_i \bar{n}_i^P + dq - c_i \right\}$. These subset are closed as their complementary $n^O$ in $[0,1]$. Each term defining $n^P$ and $n^{OS}$ is continuous on $\bar{n}_i^P$ and $\bar{n}_i^{OS}$, which makes also continuous $n^P$ and $n^{OS}$, then $n^P$ on $\bar{n}_i^P$ and $\bar{n}_i^{OS}$, then the correspondence $T'$ and $T$. These transformations being defined from a compact set on itself, they admit at least a fixed point. This (these) fixed point(s) are Nash equilibrium (equilibria) of the second step sub-game. Suppose that $\left\{ n_{P}^{*}, n_{OS}^{*}, n_{O}^{*} \right\}$ is a (one of the) fixed point(s) of this transformation. The size of the population of developers adopting the open source software $n_{D}^{OS}*$ can then be deduced elementarily from the relation of definition $n_{D}^{OS} = \inf \left[ n_{D}^{OS}, \mu \right]$.

Proof of Proposition 2: Let $(q^+, c_i^+)$ an economically viable pair of quality and licence cost of the proprietary software. When there is no OSS, since agents are not ex-ante differentiated in their access to proprietary software, if $n_{P}^+, \left( 0 \leq n_{i}^+ \leq 1 \right)$ is a Nash equilibrium of the second step sub-game, by definition, $\left( \alpha_i n_{P}^+, dq^+ - c_{i}^+ \geq 0 \right) \Rightarrow \left( \alpha_i + dq^+ - c_{i}^+ \geq 0 \right)$ from which we deduce that $n_{P}^+ = 1$ is also a Nash equilibrium of the second step sub-game for the economically viable pair $(q^+, c_i^+)$. Now consider for this same pair $(q^+, c_i^+)$ the presence of an OSS and suppose that the expected distribution of agents is given by
Derivation of the Second Step Sub-Game Nash Equilibria: The possible outcomes of the second step sub-game can be described exhaustively, according to the values of parameters $\alpha_i$, $\alpha_\mu$, $c^{EU}$, $\mu$ and $d$, and the first step determined variables $q$ and $c_i$. For each set of values of these parameters and variables, we simultaneously use two related figures which capture the role of expectations on the individual utilities associated with the three terms of the choice (adopting the proprietary software, the OS software or choosing reservation).

On the upper chart, we display the utility of the $i^{th}$ agent respectively as P-user and OS-user while agents $j, j \in [0,i]$ adopt the OS software and agents $k, k \in [i,1]$ after her adopt the proprietary one (we designate agent $i$ as the "marginal agent"). Intersections determine interior solutions, i.e., equilibria such that one part of agents adopts OS and the other one the proprietary software. The choice of the proprietary software by all agents is always a solution when $u^{OS}_i$, i.e. the utility of the marginal agent as P-user is non-negative when $i = 0$. The choice of the OS by all agents is also a solution when the utility $u^{OS}_i$ of the marginal agent when, as $i = 1$, is such that $u^{OS}_i \geq u^{P}_i$. The equation corresponding to the utility generated by the OS software for the marginal agent is $\left(\alpha_i + \frac{\alpha_i d}{\mu}\right)$ if $i \in [0,\mu]$ and $\left(\alpha_i + \frac{\alpha_i d}{\mu}\right)$ if $i \in [\mu,1]$. As a consequence, the slope of $u^{OS}_i$ depends on the rank of this agent: this slope is $\left(\alpha_i + \frac{\alpha_i d}{\mu}\right)$ as $i \in [0,\mu]$ and $\left(\alpha_i + \frac{\alpha_i d}{\mu}\right)$ as $i \in [\mu,1]$. According to the sign of this last term, we are able to distinguish two series of cases: i) cases where the adoption costs only dampen the effect of adoption externalities and ii) cases where the opposite relation is verified (adoption costs are higher than adoption externalities for the end-user). The curves $u^{P}_i$ and $u^{OS}_i$ may exhibit no, one or two intersections within the interval $[0,1]$. Let us first consider the case of a single intersection. In this case, the intersection of the curves $u^{P}_i$ and $u^{OS}_i$ corresponds to the critical level of adoption of the OS software sufficient to induce the adoption of the OS software. Supposing that all agents $j \in [0,i]$ adopt the OS software, this intersection indicates that the $i^{th}$ agent indifferently adopts the OS software or the proprietary one and the subsequent agents will prefer the proprietary software.

Figure 1b provides the expected size of the OS software users’ population as a function of its expected level. Given A1, $\forall i, u^{OS}_i \geq u^{OS}_i \geq 0$. As a consequence, A1 excludes that in any case, $\bar{n}^{OS} \neq 0$. Hereby, $\bar{n}^{OS} = 1 - \bar{n}^{OS}$ and the transformation $n^{OS} = t^{OS}(\bar{n}^{P},\bar{n}^{OS})$ can be expressed as single variable function of $\bar{n}^{OS}$. It is then the representation of the transformation $n^{OS} = t^{OS}(\bar{n}^{P},\bar{n}^{OS})$ that Figure A-1b depicts. According to A2, agents are supposed to adopt the OS software in the reverse order of their adoption costs (end users endowed with the lowest adoption costs, adopt first). Thus, the intersections of this function with the 45° curve can be regarded as levels the OS population such that if they are expected by the population, they are also confirmed by their rational choices; these positions are Nash equilibria of the second step sub-game.
Figures 1a and 1b can be related in a simple way: as long as the \( i^{th} \) agent does not wish to adopt the OS software when agents \( j \in [0, i] \) are expected to have adopted it, the function associating the desired level of adoption with the expected level one remains below the 45° curve. The 45° curve is cut up on figure 1b with a vertical intersection, since the (first) intersection of \( u_i^p \) and \( u_i^{OS} \) is attained on figure 1a in an increasing part of function \( u_i^{OS} \) before the point \( i = \mu \). This intersection defines a separating Nash equilibrium E3, in the sense that the agents with low adoption costs adopt OS in E3 while the others use the proprietary software. After this intersection, the slope of the function \( u_i^{OS} \) is still positive, even after \( i = \mu \). There then exists for a certain value of \( i \), a second level of OS users such that if the size of OS users is expected by agents greater than or equal to this level, all agents choose to adopt OS. In this case (where the second equilibrium of section 3.2. rules as well), the curve connecting the OS adoption level to the expected OS adoption level has the form depicted in figure 1b. There is a Nash equilibrium E2 at the NE corner of the box. At last, when no agent is supposed to adopt OS, the equilibrium E1 of Proposition 2 is at the SW corner of the box.

The local stability of the Nash equilibria can also be analyzed in the case of static expectations with the help of figure 1b. Suppose thus that agents expect the size of the OS software users to be somewhere between equilibria E1 and E3. Then, their answer will be for all to adopt the proprietary software. We conclude that all initial expectations of a relatively low level of OS software users rapidly drive to a Nash equilibrium where the OS software is fully crowded out (E1). Now suppose now that the initial expected size of the OS software users is somewhere between E3 and E2. Then, agents declaring choosing the OS software are more numerous than expected; some of the agents having initially chosen to be proprietary software users change their strategy and join agents declaring choosing the OS software. By successive iterations, the number of the OS software users increases until reaching equilibrium E2, where only the OS software remains.

Other cases are possible, according to the values of parameters and first step variables. They are listed on figures 2 to 7. Note that the assumption that the proprietary software is economically viable excludes all case where the curve \( u_i^p \) of the marginal agent has no positive intersection with the ordinate axis, excluding, jointly with A1, the case where there is one single equilibrium with no adoption.
Case 1
Condition of appearance :
\[
\begin{align*}
\alpha_i (1 - \mu) &> \bar{c}^{EU} \\
 dq - c_i &> -\alpha_i \\
 dq - c_i &< \alpha_i (2\mu - 1) + d\alpha_z
\end{align*}
\]

Case 2
Condition of appearance :
\[
\begin{align*}
\alpha_i (1 - \mu) &> \bar{c}^{EU} \\
 dq - c_i &> \alpha_i (2\mu - 1) + d\alpha_z \\
 dq - c_i &< \alpha_i + d\alpha_z - \bar{c}^{EU}
\end{align*}
\]

Equilibria of the second step sub-game: cases 1 to 3.
Case 4

Condition of appearance:

\[
\begin{align*}
\alpha_i (1 - \mu) &< \bar{c}^{EU} \\
dq - c_i &> -\alpha_i \\
dq - c_i &< \alpha_i (2\mu - 1) + d\alpha_z \\
dq - c_i &< \alpha_i + d\alpha_z - \bar{c}^{EU}
\end{align*}
\]

Case 5

Condition of appearance:

\[
\begin{align*}
\alpha_i (1 - \mu) &< \bar{c}^{EU} \\
dq - c_i &> \alpha_i (2\mu - 1) + d\alpha_z \\
dq - c_i &< \alpha_i + d\alpha_z - \bar{c}^{EU}
\end{align*}
\]

Case 6

Condition of appearance:

\[
\begin{align*}
\alpha_i (1 - \mu) &< \bar{c}^{EU} \\
dq - c_i &> \alpha_i (2\mu - 1) + d\alpha_z \\
dq - c_i &> \alpha_i + d\alpha_z - \bar{c}^{EU}
\end{align*}
\]

**Equilibria of the second step sub-game: cases 4 to 6.**
Case 3

Condition of appearance:

\[
\begin{aligned}
\alpha_i (1 - \mu) &< \bar{c}^{EU} \\
dq - c_z &> -\alpha_i \\
dq - c_z &< \alpha_i (2\mu - 1) + d\alpha_i \\
dq - c_z &> \alpha_i + d\alpha_i - \bar{c}^{EU}
\end{aligned}
\]

Equilibria of the second step sub-game: case 7.
Appendix 2: Nash equilibria of the first step sub-game

Proof of Proposition 3:

Whatever \( (\alpha, \alpha, d, \mu, \overline{c}^{EU}, \beta) \), when \( c_L \to +\infty \), as the quality \( q \) is limited to \( q = 1 \), the demand of the proprietary software by users vanishes and the profit becomes negative. If a maximum of (4) exists, it is then always attained for finite values of the two control variables \((c_L, q)\) of the firm. Consider then the profit defined by expression (4). This expression depends directly on a continuous way on arguments \((c_L, q)\). It depends also indirectly on the same arguments by the effect of \( n^{***}(c_L, q) \) that appears directly and indirectly in \( p'' = \left(1 - n^{***}\right) \). The form of (4) attests that when the second step sub-game has 3 equilibria, the continuity of the two moving ones on \((c_L, q)\) is a sufficient condition of the continuity of (4) on the same arguments. The equilibria of the second step sub-game are the fixed points of the correspondence \( T \) defined in the proof of Proposition 1 (see Appendix 1). \( T \) is generated by the intersection of \( \alpha_i n^{OS} + \left(\alpha_i d \overline{n}^{OS} - c_i^d\right) \) when \( i \) ranges from 0 to 1. Only the first term depends continuously on \((c_L, q)\): the intersection moves then continuously on \((c_L, q)\). The correspondence \( T \) is then continuous on the arguments \((c_L, q)\) and its fixed points are also continuous on the same variables. We deduce that the expression (4) is continuous on \((c_L, q)\). As 0 in a low bound of the profit, \((c_L, q)\) are both defined on a compact, and \( T \) admits at least one maximum on this compact, which proves Proposition 3. An illustration of the continuity of \( n^{***}(c_L, q) \) and \( n^{***}(c_L, q) \) is given by considering the cases analysed in the end of Appendix 1. Consider for instance the values of the parameters \((\alpha, \alpha, d, \mu, \overline{c}^{EU})\) and arguments \((c_L, q)\) defined by the conditions of cases 4 and 5. The conditions \( \alpha_i(1 - \mu) < \overline{c}^{EU} \) and \( dq - c_L < d\alpha_i - d\alpha_i - \overline{c}^{EU} \) are joint conditions for these two cases. The transition from case 4 to case 5 correspond to the change of \(-\alpha_i < dq - c_L < \alpha_i(2\mu - 1) + d\alpha_i \) into \( \alpha_i(2\mu - 1) + d\alpha_i < dq - c_L \) with the intermediate case not represented \( \alpha_i(2\mu - 1) + d\alpha_i = dq - c_L \). From (4) to (5) it then appears that the correspondence represented by figures 4b and 5b moves continuously, with a progressive shift of \( n^{***}(c_L, q) \) from the left side of \( \mu \) to the right side of the same proportion of users. Similar continuous transitions can be analysed for instance from case 1 to case 2 or from 4 to case 7, illustrating in this last case the continuous shift of \( n^{***}(c_L, q) \) and \( n^{***}(c_L, q) \) with \((c_L, q)\).

Appendix 3: Market Failures and Welfare Properties

Proof of Proposition 4:

Suppose that the optimal quality / price pair \((q^*, c_L^*)\) involves case, the occurrence of two stable equilibria, \( E_1 = \{n^*_1 = 1, n^*_d = 0, n^*_0 = 0\} \) and \( E_2 = \{n^*_d = 0, n^*_0 = \mu, n^*_1 = 1\} \). At \( E_1 \), individual utility of users is uniform and given by \( u^*_i = \alpha_i + dq^* - c_L^* \) for the developers and \( u^*_i = \alpha_i + \alpha_d d - \overline{c}^{EU} \) for the "last" end-users. For developers, the result of the comparison between the two equilibria is such that \( E_1 \equiv D \equiv E_2 \) when \( \alpha_i \leq q^* - (c_L^*/d) \) and \( E_2 \equiv D \equiv E_1 \) when \( \alpha_i \geq q^* - (c_L^*/d) \). If \((q^*, c_L^*)\) is an optimal solution for the firm, the expected profit is non-negative and \( u^*_i = \alpha_i + dq^* - c_L^* \geq 0 \). For the "last" end-users, the result of the comparison is \( E_1 \equiv E_2 \) when \( \alpha_i \leq q^* - \left(\overline{c}_L^* - \overline{c}_i^{EU}\right) / d \) and \( E_2 \equiv E_1 \).
when \( \alpha_z \geq q^* - \frac{(c_z^* - c_{EU}^*)}{d} \). One concludes that the "all proprietary" equilibrium \( E_1 \) is preferred by all users (and the firm) when \( \alpha_z \leq q^* - \frac{(c_z^*)}{d} \) and that there exists conflict of interest in the other cases. 

**Proof of Proposition 5:** Suppose that the optimal quality / price pair \((q^*, c_z^*)\) involves case, the occurrence of two stable equilibria, \( E_1 = \{n^* = 1, n_{OS}^* = 0, n_{OS}^* = 0\} \) and \( E_3 = \{n^* = 0, n_{OS}^* = 0, n_{OS}^* = \mu, n_{OS}^* = \eta\} \) with \( \mu < \eta < 1 \) in the linear-quadratic. At \( E_1 \), individual utility of users is uniform and given by \( u_i^p = \alpha_i + dq^* - c_z^* \). At \( E_3 \), utility is \( u_i^p = \alpha_i \eta + \alpha_i d - \frac{\mu}{(1-\eta)^2} \), \( \mu < \eta \) for the "first" end-users, and \( u_i^p = \alpha_i (1-\eta) + dq^* - c_z^* \), \( \eta \leq 1 \), for the "last" end-users. For developers, the result of the comparison between the two equilibria is such that \( E_1 \) is preferred by all users (and the firm) when \( \alpha_z \leq q^* - \frac{(c_z^*)}{d} \) and that there exists conflict of interest in the other cases.

**Proof of Proposition 6:** Consider the pair \((q^*, c_z^*)\) corresponding to the optimal price / quality solution when users have no other alternative than adopting or not the proprietary software. This solution is such that the firm takes the whole surplus and that users' utility vanishes, \( i.e. \forall i, u_i^p = \alpha_i + dq^* - c_z^* = 0 \). The profit is then \( \pi = \sum \alpha_i \sum c_z^* - \beta q^* \). If \((q^*, c_z^*)\) represents the optimal price / quality solution when users have the choice between the OS and proprietary software and if \( \pi (E_1) = c_z^* - \beta q^* \), the profit of the firm in the case where the second step equilibrium is \( E_1 \). Suppose that \( \pi (E_1) = c_z^* - \beta q^* \), then \( c_z^* = \frac{\beta q^*}{\mu} \). Then, \( c_z^* = \frac{\beta q^*}{\mu} \). One concludes that agent \( i = 0 \), \( i.e. \) the first developer. At \( E_1 \), the utility of this agent as an OS-user is \( u_i^p (E_1) = 0 \) in such a way that this agent is indifferent between the two technologies. For expected amount of \( n_{OS}^* \) larger that 0, this utility is \( u_i^p (E_1) = \alpha_i d - c_z^* \). One concludes that agent \( i = 0 \) will choose OS against the proprietary software, except when \( n_{OS}^* = 0 \). It is then excluded that \( E_1 \) will be a locally stable equilibrium if \( (q^*, c_z^*) = (\bar{q}, \bar{c}_z) \) and \( \pi (\bar{q}, \bar{c}_z) = 0 \) which disqualifies \((\bar{q}, \bar{c}_z)\) as an equilibrium solution. One conclude that \( E_1 \) is always such that \( \forall i, u_i^p (E_1) = \alpha_i + dq^* - c_z^* > 0 \), that demonstrates proposition 6.