Open Source, Dual Licensing and Software Competition

Éric Darmon
CREM UMR CNRS 6211, University of Rennes 1, France

Dominique Torre
CREDEG CNRS, Université Nice Sophia-Antipolis, France

March 2014 - WP 2014-05
Open Source, Dual Licensing and Software Competition

Eric DARMON · Dominique TORRE

March 2014, 12th

Abstract To distribute software, commercial vendors of proprietary software have the opportunity to use some dual licensing (DL) strategy i.e. to provide their software under two different licensing terms (proprietary and open source). We investigate the relevance and impacts of this distribution strategy in the presence of an incumbent open source software competitor. We determine the conditions for this strategy to be profitable for the commercial firm and its impact on price, market shares and welfare. We show that dual licensing may be used as a complement for proprietary software when development spillovers are large. We examine how, in this case, a dual licensing strategy can be used to exclude the open source software from the market and how this is compatible with higher price and lower market share for the proprietary distribution. This situation can also generate conflicts of interests between proprietary software and users resulting in sub-optimal outcomes. Finally, our analysis reveals the key role played by development spillovers and software compatibility for the DL decision. (JEL Classification: D23, D42, L86)

Keywords dual licensing · hybrid business model · software distribution strategy · open source · spillover

1 Introduction

Open Source Software (OSS) emerged in the late 1970s and represent a major shift in the software industry which is now diffusing to other industries in the form of the Open Science and Open Innovation models (David (2004), Chesbrough (2006)). Because OSS emerged in reaction to closed...
source software\textsuperscript{1}, these two models of software production and distribution have often been opposed: different types of network externalities (e.g. prominent role of development spillovers in the case of OSS), different organizational features (e.g. top-down vs bottom-up mechanisms), different distribution strategies (e.g. upgrade and release strategies) and finally different incentives to innovate and different welfare impacts (See e.g. Schwarz and Takhteyev (2010)). In the context of these conflicts, the focus when analyzing the relationship between OS and proprietary software has been first put on competition (see e.g. Dalle and Jullien (2003), Bonaccorsi and Rossi (2003), Casadesus-Masanell and Ghemawat (2006), Lanzi (2009), Darmon et al. (2011), see also Arora and Bokhari (2007) for an analysis based on industry dynamics) together with welfare issues (see e.g. Scotchmer (2010)). In comparing the two models, some papers have investigated the conditions for a proprietary software provider to “go open” and switch to a completely new model based on open-source terms (see Parker and Van Alstyne (2005) and Economides and Katsamakas (2006) in the case of platform software).

However, the nature of the link between the OS and Closed source paradigms has evolved over time and we can see some convergence between the two models as exhibited by the emergence of so-called “hybrid business models”\textsuperscript{2}. On the one hand, many commercial players now use OSS as the basis for derived cross-products or services (see e.g. Redhat in the case of Linux, Canonical Ltd in the case of Ubuntu) while on the other hand, some players that traditionally sell closed source software are choosing to open up parts of their code and get closer to OS communities\textsuperscript{3}. For these players, the incentives to do so can vary. They may want to increase the feedback from developers to enable better targeted R&D efforts for future developments (better identification of user needs, etc.). They might want to partly ‘outsource’ their R&D effort in order to save on some of these costs (cf. Llanes and de Elejalde (2013), Kumar et al. (2011) for a theoretical analysis and Fosfuri et al. (2008) for an empirical study). In relation to users, the aim may be simply to increase the installed user base. Since license terms play a key role in defining the scope of the Intellectual Property Rights protecting software, this convergence is reflected in the licensing terms used by firms (and even by some firm-driven communities) to distribute their software. One way to implement an “hybrid” strategy is to adopt a dual (or even multiple) licensing strategy. Dual licensing (DL) consists of delivering software under two different license terms, one being more ‘open’ than the other (cf. Välimäki (2003) for a presentation of several case studies)\textsuperscript{4}. Closest to our setting, Comino and Manenti (2011) examine the incentives for a monopolist firm to employ a DL strategy and show the prevalence of the development spillover in the DL decision. However, this decision is not independent on the nature and intensity of competition on the software market before DL introduction. For instance, a DL strategy may be a reaction to the market entry of an OS player (and without this may not have been introduced). Hence, models based on DL in the monopolistic case cannot investigate such strategic interactions. Also, our work is novel in considering competition not between two profit-motivated actors but between a profit-driven actor and an OS community generating original types of interactions in the context of Dual Licensing.

\textsuperscript{1} See e.g. Bitzer and Schröder (2006) for a survey on the emergence and recent history of the Open Source movement. See also Lerner and Tirole (2005) for a general presentation of economic issues related to the Open Source trend.

\textsuperscript{2} See Dahlander and Magnusson (2008) for a review of hybrid business models, see also Campbella Kelly and Garcia Swartz (2010).

\textsuperscript{3} See e.g. http://robertogaloppini.net/2008/03/20/open-source-at-microsoft-an-analysis-ofmicrosoft- open-source-strategy/ for some anecdotal illustration about Microsoft. See also Rolandsson et al. (2011) and Casadesus-Masanell and Llanes (2011)

\textsuperscript{4} Such strategy should not be confused with freeware and shareware distribution strategies in which one software is distributed for free but is available with a limited set of functionalities of for a limited time period. In both cases, the software source code is not being opened (see Haruvy and Prasad (2005) for an economic analysis of this distribution strategy).
In this paper, we introduce competition between a proprietary software vendor (hereafter \( P \)) and an OSS (OS). For that, we design an original model in which the vendor has the opportunity to introduce a hybrid version of its software through DL (hereafter \( OSP \)). Our model captures the most relevant interactions between the different software and analyse in depth the consequences on price, market shares and welfare. We determine the conditions for this strategy to be profitable for the commercial firm and its impact on price, market shares and welfare. We first show also how a dual licensing strategy can be used to exclude the open source software from the market and how this is compatible with higher price and lower market share for the proprietary distribution. This strategy is however profitable only when development spillovers are large. Second, we show that dual licensing may be used as a complement for proprietary software. This situation also reveals some conflicts of interests resulting in sub-optimal outcomes. As well as Comino and Manenti (2011), we find back the key role played by development spillovers. Introducing competition enables us to stress the role of software compatibility on the DL decision also. I also enables to fully study the complementarity and substitution effects between all software.

Section 2 develops the model and characterizes the potential outcomes in the benchmark case. Section 3 introduces the DL and characterizes the potential equilibrium outcomes. Section 4 discusses the conditions for DL to be profitable and analyzes its effect on price, market shares and welfare. Section 6 discusses the results and concludes.

2 The Model

We consider the competition between two platform softwares using an Hotelling approach. Platform software (further simply software) is software that user-developers exploit to build their own pieces of software. As opposed to end-users, Platform software users that we consider then are user-developers a la Von Hippel (cf. e.g. von Hippel (2007), von Hippel (2001)) i.e. they are experienced users able to adopt software and also to contribute to the development of new pieces of software (as opposed to final users which adopt software but do not contribute to code development). These user-developers can choose between a (platform) software released under a GPL license produced by an OS community of developers and software produced by a commercial (i.e. profit motivated) vendor.

2.1 The software vendor

In the benchmark case, the commercial vendor only distributes this software under proprietary license terms (i.e. closed source code, or \( P \)) and sells it at price \( p \) to obtain revenues\(^6\). In the benchmark case, the vendor distributes this unique version. DL enables distribution of another version of its software released under partially open license terms. For simplicity, we assume that the original software is produced at no cost. In our context, this assumption is highly consistent with the actual cost structure in this industry\(^7\). The vendor maximizes its profit with respect to the

---

\(^5\) For a formal definition of General Public License (GPL), see http://www.gnu.org/licenses/gpl-3.0.html.

\(^6\) Without any effects on our model, this price can take diverse forms in practice: one-shot license, royalties charged for re-use or re-distribution of the software, etc.

\(^7\) Since software production costs are essentially fixed costs, this is a common assumption for software markets or markets for digital goods (cf. e.g. Shapiro and Varian (1998)). Introducing such costs would simply introduce a minimum scale below which the commercial vendor would not be operate and would not change our results qualitatively.
license cost of the proprietary software, and given the above simplifications this profit ($\pi$) is simply the vendor’s revenues. Thus, if $m_p$ denotes the number of adopters of the proprietary software, this profit reduces to $\pi(p) = m_p p$.

2.2 User-Developers

There exists $m$ user-developers (or ‘users’), each of them may potential adopt a software. Without loss of generality, the total number of users is normalized to 1 (i.e., $m = 1$). Because each user has its own software project, each one has different needs regarding software features and requires specific type of software to fulfill these needs. These differences require the developer to write additional pieces of code to adapt the original software to specific uses.

To depict this type of heterogeneity, let us assume that users are uniformly distributed on a unitary segment. Writing additional code requires development effort which we can depict as an additional development cost associated with the adoption of one software. This cost is increasing with the distance to the location of the software on the segment: the less a software fulfills the user’s needs, the more this user will be required to write specific modules to adapt the software to his needs and the higher will be its development cost. Let us assume that the proprietary software is located at location 0 on the unitary segment while the OS software is located at location 1, opposite to the proprietary software.

We can thus simply write the utility $U_p(i)$ of the proprietary software derived by User $i$, ($i \in [0, 1]$) when using the proprietary software with $U_p(i) = v - ai - p$ where $v$, $ai$ and $p$ respectively denote the benefits derived from the P software, the additional development cost ($a > 0$) associated with this software, increasing with the location $i$ of User $i$, and the fees paid by the users of the proprietary software to the vendor.

In the case of the OS software, users need to write additional lines of code to develop new pieces of software. This translates into an additional development cost $\beta(1 - i)$ (with $\beta > 0$). However, because it is OS, there is no direct cost for using the software. The utility $U_{os}(i)$ derived from OS adoption is then $U_{os}(i) = u - \beta(1 - i)$ where $u$ depicts the benefits derived from the OS software.$^9$

2.3 Outcomes in the Benchmark Case

In the general model, the vendor plays first and selects a licensing and pricing strategy. In the benchmark case, since DL is not introduced, this decision rests on price. Observing software licensing terms and conditions, potential users play second and choose to adopt (or not) one among the

$^8$ There are two rationales for this assumption. First, from a historical point of view, OS projects often developed in reaction to proprietary standards because closed-source software was targeted to specific needs (generally the most common ones) but did not meet all the developers’ needs. Second, because of its technical and commercial aspects, software cannot satisfy exactly the same needs and differentiate in multiple dimensions. For modeling purpose, it is here sufficient to consider that this differentiation is one-dimensional, which makes it reasonable to assume that the two softwares are located opposite one another.

$^9$ Note that unlike a common assumption in this literature, we do not consider here the potential spillovers of OS user-developers on this software. We do so because our focus here is on the implication of dual licensing on both OS and P software and we introduce an externality to depict this particular effect. Adding a second type of spillovers (e.g. a development spillovers of OS on itself) would reduce tractability with no substantial benefit for understanding on the issue analyzed in this paper.
different available pieces of software. This defines a two step sequential game with perfect information that can be solved by backward induction. In the benchmark case, an equilibrium situation is fully characterized by an optimal price \( p^* \), and a distribution of users among the two software products. Let \( m^*_p \) and \( m^*_os \) denote the respective optimal market shares of P and OS (software) users and of OS users (with \( m^*_p + m^*_os \leq 1 \)).

Since we are primarily interested in initially competitive outcomes, we focus on situations where the two softwares (OS, P) are coexisting before the vendor may choose to adopt DL. Depending on the parameters, two situations can occur in the benchmark case: one with a full adoption (i.e. fully served market, Lemma 1) and another with partial adoption (i.e. non fully served market, Lemma 2).

**Lemma 1** Initial Full Adoption [P-OS]. When \( v > u - \beta \) and if \( 2\alpha + \beta < v < u + 2\alpha + \beta \) or if \((2\alpha + \beta)(\beta - u)/\beta < v \leq 2\alpha + \beta \), then all user-developers adopt one of the two software products. The commercial license is sold at price \( p^* = (v + \beta - u)/2 \), the optimal profit of the commercial vendor is equal to \( \pi^* = (v + \beta - u)^2/4(\alpha + \beta) \) and the market share of the P software is \( m^*_p = (v + \beta - u)/2(\alpha + \beta) \) while that of the OS software is \( m^*_os = (u - v + 2\alpha + \beta)/2(\alpha + \beta) \). Proof: see Appendix 1 ■

**Lemma 2** Initial Partial Adoption. [P-⊘-OS]. If \( v < 2\alpha \) and \( \beta > 2\alpha u/(2\alpha - v) \), then a fraction \( m^*_p = v/2\alpha \) of all potential users adopts the proprietary software, a fraction \( m^*_os = u/\beta \) adopts OS software and the remaining part does not adopt any software products. The commercial license is sold at price \( p^* = v/2 \) and the optimal profit of the vendor is then \( \pi^* = v^2/4\alpha \). Proof: see Appendix 1 ■

The conditions on parameters that define both outcomes are standard w.r.t. the literature on horizontal differentiation and relate to the magnitude of the development cost w.r.t. the intrinsic utility brought by the two software. Since these two sets are mutually not compatible, this rules out potential multiple equilibria in the benchmark case.

### 3 Equilibrium Outcomes with Dual Licensing

Through the introduction of an alternative software (OSP) derived from its regular software (P), the vendor may introduce DL if it finds it is profitable to do so. We refer to this hybrid software as OSP since it comprises some of the properties of both OS and P software. The introduction of the OSP software has several effects on the utility of the different software products. First, the adopters of the OSP software may benefit from the functionalities \( v \) developed for the proprietary software but without paying the license cost attached to them. Similarly to the P software, they incur an additional development cost \( \alpha_i \) depending on their location \( i \) in the segment. Hence, the licence price does not enter into the indirect utility when adopting from the OSP.

Second, thanks to the “open” source license terms under which the OSP software is released, some users will develop using this software (by adding new lines of codes, creating new modules) and will release them to the community of OSP users. By defining specific license terms, these lines of code may be partly appropriated by the vendor and be reintegrated to improve the P software.\(^{10}\)

\(^{10}\) For that, the vendor has to use some types of license that allows for further use/distribution without the obligation to disclose all the derived code publicly. Notably, BSD (Berkeley Software Distribution) or LGPL (Lesser General Public License) allows for such re-use. The vendor may also define any ad hoc license terms.
Thus, this generates a spillover from the OSP to the P software that will be captured by Parameter $a$. Note that this term may also be interpreted as some level of openness in the hybrid license. Higher values of $a$ then mean that the license is “less open” (regarding the obligation to publicly disclose any source code derived from the software) and that the vendor is better able to privatize part of users’ innovations.\footnote{Note that the source code on the OSP software is defined “open” and so Parameter $a$ does not affect users’ ability to innovate on the OSP software. It only affects how much of this effort the vendor is able to capture. See discussion in Section 5.}

Third, the OS and OSP benefit mutually through a decrease in the development cost required for adoption of these software products. Indeed, part of the development effort of OS users may be exploited by OSP users allowing more efficient development and conversely. Since this positive effect diffuses through a reduction in development costs, it applies to the transport cost in the indirect utility of an Hotelling-based approach. Yet, the magnitude of this last effect depends crucially on the degree of compatibility between the two software products. The more compatible the two software products, the easier will be the exchange of ideas and code across them. However, if the two software products are completely non-compatible, exchanges across software products will never be possible. Let us capture the compatibility degree by parameter $\lambda$ (with $0 \leq \lambda < 1$). Two points are worth noting: (i) because the source code of the P software is closed, P users do not benefit from this effect. Hence, this effect does not account in their indirect utility; (ii) An increase in $\lambda$ has several effects. When $\lambda$ tends to 1, the two software products tend to full compatibility. In this case, developers on one of the two software products can easily grasp some “ideas” (or some functionalities) from one software and implement them on the other software thus saving on development costs\footnote{Using Fershtman and Gandal (2011)’s terminology, such cross-exchanges define a “project” spillover.}. In contrast, when $\lambda$ decreases and tends to 0, the two software products are increasingly less compatible and such cross-exchanges become less feasible, so development costs tend to those incurred in the benchmark case. In our model, this impacts on product substitutability. A decrease in $\lambda$ makes both software (OSP and P) more attractive w.r.t. P software (all things else equal). At the same time, this decrease makes the location-based differentiation between the OS and OSP software products vanish. When deciding to DL, all these potentially conflicting effects are at stake at the same time, and so our modeling is able to integrate all these effects in a single framework.

Introducing these elements, we can reformulate the three indirect utilities as follows ($a > 0$):

\begin{equation}
\begin{align*}
U_p(i) &= v - \alpha i + am_{osp} - p \\
U_{osp}(i) &= v - \alpha (1 - \lambda) i \\
U_{os}(i) &= u - \beta (1 - \lambda) (1 - i)
\end{align*}
\end{equation}

with $a > 0$ and where $m_{osp}$ is the proportion of the potential users adopting the hybrid software (thus we have $m_p + m_{osp} + m_{os} \leq 1$).

**Lemma 3** When the OSP software is active ($m_{osp} > 0$), the population adopting this software is always adjacent to the population adopting the proprietary software on the unit segment. Proof: see Appendix 2

Lemma 3 applies to all potential equilibrium outcomes with DL and depicts the location of OSP w.r.t. to P and OSP on the Hotelling line. Again, we concentrate on some parameter configurations where OS and P are both active in the benchmark case (i.e. before DL is introduced). Propositions
1, 2 and 3 highlight the three possible types of equilibrium outcomes with DL. We focus the discussion on the role played by $a$ in defining these outcomes.

**Proposition 1** Full Adoption with Three Active Software [P-OSP-OS]. Whatever the value of $a$ and under some conditions for other parameters, there exists fully served equilibrium outcome where some users adopt the OS software, some other adopt the OSP software and remaining ones adopt the P software (with $m^*_{os} = (a + 2a\lambda)(u - v + \beta(\lambda - 1))/2(\alpha + \beta)(\lambda - 1)(a + \alpha\lambda)$; $m^*_{osp} = (v - u + \alpha(\lambda - 1))/2(\alpha + \beta)(\lambda - 1)$ and $m^*_p = a(u - v + \beta(\lambda - 1))/2(\alpha + \beta)(\lambda - 1)(a + \alpha\lambda)$). The optimal equilibrium price charged for the P software is $p^* = a(u - v + \beta(\lambda - 1))/2(\alpha + \beta)(\lambda - 1)$ and the profit $\pi^* = a^2(v - u + \beta(1 - \lambda))^2/(4(\alpha + \beta)^2(\lambda - 1)^2(a + \alpha\lambda))$. Proof: see Appendix 2.

Proposition 1 depicts an equilibrium with full adoption where all users adopt one software and each software get some users. Interestingly, the existence conditions for such equilibrium do not impose any restriction on the level of the spillovers from the OSP to the P software, as captured also measures the ability of the commercial vendor to incorporate development spillovers from the dual version into the proprietary software. This has a practical implication: implementing a DL strategy does not depend here on the magnitude of the spillovers (or equivalently whatever the degree of openness of the OSP license). However, the price charged and therefore the profit, are both positively affected by $a$: the vendor can charge a higher price when successfully incorporate many of the features from the OSP and this increases the vendor’s profit. Note however that this result does not mean that the DL strategy is always profitable as compared to the situation without DL (see Section 4).

**Proposition 2** Partial Adoption with Three Active Software [P-OSP-2-OS]. Whatever the value of $a$ and under some conditions for other parameters, there exists non fully served equilibrium outcome where some users adopt the OS software, some other adopt the OSP software, some others adopt the P software and remaining ones do not adopt from any software (with $m^*_{os} = u/\beta(1 - \lambda)$; $m^*_{osp} = v(\alpha + 2a\lambda)/(2\alpha(\lambda - 1)(a + \alpha\lambda))$ and $m^*_p = av/(2\alpha(\lambda - 1)(a + \alpha\lambda))$ and $m^*_p + m^*_{osp} + m^*_{os} < 1$). The optimal equilibrium price charged for the P software is $p^* = av/2\alpha(1 - \lambda)$ and the profit $\pi^* = a^2v^2/(4\alpha^2(\lambda - 1)^2(a + \alpha\lambda))$. Proof: see Appendix 2.

Proposition 2 states that a non fully served market is compatible with DL introduction. As well as in the benchmark case, this equilibrium is likely to arise when development costs associated with the different software products are relatively large. Similarly to Proposition 1, spillovers ($a$) play essentially the same role as that played in the case of a fully served market i.e. this outcome may occur whatever the magnitude of the spillovers but both price and profit increase as $a$ increases.

**Proposition 3** Full Adoption with OS Crowding Out [P-OSP]. Whatever $a$ and under some conditions for other parameters, an equilibrium outcome exists where some users choose the P software ($m^*_p = a^2/(a + \alpha\lambda)$ and other users adopt the OSP software ($m^*_{osp} = 1 - a^2/(a + \alpha\lambda)$). In that case, the OS software is not used ($m^*_{os} = 0$). The P software is sold at price $p^* = a/2$ and the profit of the vendor is $\pi^* = a^2/4(a + \alpha\lambda)$. Proof: see Appendix 2.

In this last outcome, the (indirect) effect of introducing DL is to crowd out the OS software. This situation occurs only when development costs on the OS software are higher than those on
Eric DARMON, Dominique TORRE

the other two software products. In that case, typical OS users may switch to the OSP and no user adopts the OS software after the introduction of DL.

Similarly to the two previous outcomes, the price charged by the vendor depends positively on the amount of spillovers from the OSP. However, the price here is simply equal to these spillovers and does not depend on any other parameters. In particular, the price does not depend on $v$. Indeed, the only distinctive element between the OSP and the P software is spillovers (the intrinsic utility $v$ is common to the two software). As a consequence, the main driver of the price of the P software is not the intrinsic value of that software but the amount of the spillovers generated by the OSP software.

When considering the sets of parameters defining the three outcomes, we can check that these sets are all mutually incompatible. This rules out multiple equilibria in the case with DL also.

4 Dual Licensing decision

We analyze the decision of the vendor to introduce DL and its effect on the market structure. Considering the different potential equilibria of the game (in the benchmark case and with DL), we need to consider all the potential switches raised by the introduction of DL. By considering the different equilibrium conditions of Lemmas 1 and 2, we can see that these conditions are mutually exclusive, thus ruling out multiple equilibria in the benchmark case. Similarly, the equilibrium conditions of Propositions 1, 2 and 3 are all mutually exclusive, thus ruling out multiple equilibria when DL is introduced. Taken together, these statements mean that, for a given set of parameters, the “initial” situation (i.e. prior to introduction of DL) is unique and once DL is introduced, it can lead to one “final” (with DL) equilibrium outcome only. The type of the initial and final outcomes respectively depends on the whole set of the parameters. To analyze the DL decision and its effects on price, profit and welfare, we need to consider one particular outcome with DL and study whether this outcome can be reached from different initial outcomes (without DL). We consider highlight two interesting situations.

4.1 Dual Licensing and Open Source Software Crowding Out

Consider first the situation in which DL introduction leads to OS crowding out (Proposition 3). This case raises a side issue about the value of $\lambda$ to consider in the long run. Since $\lambda$ measures the reciprocal savings on development costs allowed by the code opening of the OSP software, users of the OS and of the OSP software may both experience some positive effects in the short run. However, these positive effects are linked directly to the cross-fertilization of ideas between the OS and the OSP software. Hence, in the long run, since all OS users will have left this software, these effects are likely to vanish. Thus, in the long run, a reasonable (and conservative) approach is to suppose that $\lambda$ tends to 0. Let thus the case $\lambda = 0$ be defined as the long run position and refer the reader to Appendix 8.3 for the case $\lambda \geq 0$.

---

For completeness, note also that an equilibrium with partial adoption (i.e. where some users do not adopt) in which only the P and OSP software products are active is not possible (see Proof in Appendix 2).

Note however that the only unpossible combination is when switching to an initial fully-served outcome to an non-fully-served one. This is straightforward since DL may at most keep development costs constant.
Proposition 4  When DL introduction leads to OS crowding out, the vendor may find it profitable to DL in the long run only for a restricted set of parameters. Proof: see Appendix 3 ■

This proposition means that introducing the OSP version of the P software through DL may only sometimes be a relevant (i.e. profitable) strategy for the vendor in the long run. The initial outcome (whether fully-served or not) impacts on the conditions for DL to be profitable. From the general set of conditions, we can stress one key result about the role played by the spillovers: for DL to be profitable, the spillovers exerted by OSP on P has to be sufficiently large \((a > \max \tilde{a}, \frac{v^2}{\alpha})\). If not, the direct corollary to Proposition 4 is that DL may not be worthwhile since it results in decreased profits despite the introduction of DL may deter a “competitor” from the market. Thus, excluding an OS software by introducing DL is not always an optimal strategy and an “accommodating strategy” is preferred in this situation. This statement is especially interesting since in our model DL implies no direct cost in our model. Hence, the preference for an accommodating strategy only depends on demand-related factors and is not driven by the imposition of a specific cost structure. An important explanatory factor for the decision not to introduce DL is related to the pricing strategy. As already mentioned, when the vendor decides to introduce DL, the only difference in the utility of the OSP and the P software relates to the benefits derived from the spillovers \((\text{measured by } a)\). To differentiate the two versions, the vendor can no longer base the price on traditional determinants \((v \text{ and } \alpha)\) and the price of the OSP software therefore depends only on this element. In contrast, the price charged by the vendor in the absence of the OSP depends positively on \(v\) (and also positively on \(\beta\) and negatively on \(u\) when the market is initially fully served). When considering long-run outcomes \((\lambda = 0)\), a critical and very simple condition for dual licensing is that \(a > \tilde{a} \equiv \frac{v^2}{\alpha}\) (see Appendix 3). If the amount of spillovers \(a\) does not meet this threshold the spillovers generated by the OSP software will not be sufficient and the switch from a standard utility-based pricing strategy depending on \(v\) to some spillover-based pricing strategy (everything else being equal) will generate a loss in profit.

Proposition 5  When DL introduction leads to OS crowding out and when DL is a profitable strategy, DL negatively impacts the market share of the P software and always increases the amount of the fees paid by P users. Proof: see Appendix 3 ■

Proposition 5 explores the situation described by Proposition 4 and depicts how prices and market shares evolve after the introduction the DL when this strategy is profitable. This proposition has direct practical implications. It implies that the OSP software always acts as a substitute for both OS and P in the long run (i.e. when the benefits derived from the previous presence of the OS software vanished). It is quite clear that OSP is a substitute for OS (since all OS “core-users” i.e. those users who would preferably use the OS-software i.e. users with a location closer to \(i = 1\) will switch to the OSP software. However, we show that OSP always substitutes for the proprietary software also. Indeed, when it is profitable to introduce DL, Proposition 5 evidence some business-stealing effect in the long run. Hence, OSP is not a strict complement of P and substitutes for P for some users.\(^{15}\) Despite this decrease in market share, DL is profitable thanks to the spillover from the OSP. Since these spillovers increase the overall quality (and thus utility) of the P software, this quality improvement increases the willingness-to-pay for the proprietary version of P-core users (i.e. the WTP of those users who preferably use the P software because they are located close to \(i = 0\)). Thus, the vendor can charge a higher price for its P software even if the number of regular customers (i.e. P users) is reduced (compared to the the number of customers without DL).

\(^{15}\) It should be noted that the market share of the P software may increase in the short run (as \(\lambda > 0\), see Appendix 8.3).
4.2 Dual Licensing with an ex ante balanced situation

Let us now consider another interesting case in which the situation between the OS and the P
software is ex ante “balanced” i.e. the two software types provide the same intrinsic utility ($u = v$)
and the development costs on the two software are the same ($\alpha = \beta$). Thus, the only difference
between these software products is in the price (by definition of P and OS software). Apart from
price, this situation is thus balanced and this case thus isolates some effects from other effects based
on some exogenous differentiation ($u$ versus $v$ and $\alpha$ versus $\beta$).

**Proposition 6** Possible Outcomes with an Ex Ante Balanced Situation (Initially Fully
Served Market) When $u = v$ and $\alpha = \beta$ and in the case of initial full adoption,

(i) DL introduction is possible whatever $\lambda$ and $a$ and always leads to a P-OSP-OS outcome;
(ii) DL introduction is profitable for the vendor only when $2\beta < a \leq \beta(1 + \sqrt{3})$ and $\lambda < a(a - 2\beta)/(2\beta^2)$ or $a > \beta(1 + \sqrt{3})$;
(iii) user surplus is always increasing after DL introduction; and
(iv) DL introduction may be profitable but reduces welfare at the same time if
\[(a/(1 + \sqrt{3}) < \beta \leq a/2\) and $\lambda > (a^2 - 2a\beta)/2\beta^2 < \lambda < 1$ or $\beta > a/2$.

Proof: see Appendix 3 ■

Proposition 6 considers a situation where the market is fully served prior to DL introduction: all
users adopt one software, either the OS or the P. In these conditions, introducing DL always leads
to a situation where the market is fully served (as already noted). However, the set of outcomes
possible with DL is more limited since only a $P-OSP-OS$ outcome (fully served market with
three active software) can be achieved. Thus, starting from a balanced situation, DL can never
lead to a situation where the OS is crowded out whatever the magnitude of the spillovers. One
implication of that is that crowding out the OS software is only possible if development costs are
strictly higher with OS software compared to those incurred with P software ($\alpha < \beta$). Otherwise,
those users whose preferences closely fit to the OS software (i.e. $i$ close to 1) will always find it
profitable to use the OS software and OS deterrence is not possible, whatever the magnitude of the
spillovers.

Another interesting feature of that situation is that DL impacts on the P market share only.
Hence, when considering the decision to DL, the vendor should always expect a decrease in the
market share of the P software while the market share of the OS software is not affected. The basic
principle underlying the introduction of DL therefore, is the same as that described by Proposition
3: since the market share of the “regular” software (P) is decreasing, the only way to make DL
profitable is to increase the price charged to P core users to increase. This requires the spillovers
coming from the OSP software to be sufficiently large, as revealed by the profitability conditions.
These profitability conditions are illustrated by Figure 4.2 which shows the role of played by the
spillovers $a$ and the compatibility degree ($\lambda$). When $a$ is relatively low, DL is never a profitable
strategy. For relatively large values of $a$, DL is always profitable whatever the compatibility degree.
For some intermediate values of $a$, the situation is more subtle and DL is a profitable strategy
only when the compatibility degree is not too large. If compatibility degree is larger, code/feature
exchanges between the OS and the OSP software become easier. On the OS side, these exchanges
result in increased utility for OS users. Yet, here they do not impact on the market share of the
OS software. This neutrality can be explained here as due to the symmetry between OSP and OS
characteristics. Indeed, an increase in $\lambda$ reduces the development costs incurred on the OSP and
on the OS software by exactly the same amount since $\alpha = \beta$. Thus, the user utility from both
software increases with increased compatibility, but the position of the user indifferent between
the OS and the OSP software (describing the "balance" between these two software products) remains unchanged and as does OS market share. The effects on the OSP and the P software are more complex. On the one hand, as the degree of compatibility increases, the OSP software becomes more attractive compared to the P software (all things being equal). This increases the OSP market share to the detriment of P users. Yet, since part of the development efforts on the OSP software can be captured by the vendor (as measured by $a$), this negative effect may be counterbalanced only if $a$ is sufficiently large. The conditions for DL to be profitable illustrate this trade off between the degree of compatibility and the magnitude of the spillovers: if the level of compatibility is too high, the only way that DL will continue to be profitable is if it captures a larger fraction of OSP developers’ effort (larger $a$). If not, DL ceases to be profitable.

![Diagram](image)

**Fig. 1** The shaded area represents the values of $a$ and $\lambda$ where DL is profitable (numerical results for $u = v = 135$ and $a = \beta = 84$)

From a welfare point of view, Proposition 6 states that the user surplus is always increasing when DL is introduced. Since the total surplus simply aggregates the user surplus and the vendor’s profit, we can identify two types of situations. In some conditions (as stated by Proposition 6), both the vendor’s profit and the user surplus may increase with Dual Licensing. However, we can also identify other situations where total welfare is increasing but the vendor’s profit is decreasing. This exhibits some conflict of interests between a social planner and the software vendor. These conflicts occur when either $a$ is too low or $a$ may be larger but $\lambda$ (which has a negative effect on profit) compensates the positive effect associated with a larger $a$. Hence, a social planner needs to care about the values of $a$ and $\lambda$ in case DL distribution needs to be regulated or fostered. Note however that the switch to DL raises concern about the distribution of welfare gains within the population users depending on the adopted software. We discuss later whether this situation suggests the implementation of public policy to foster the use of DL.

**Proposition 7** **Possible Outcomes with an Ex Ante Balanced Situation (Initially Partially Served Market)** When $u = v$ and $\alpha = \beta$ and in the case of partial diffusion, DL introduction is possible whatever $a$ but only when $\lambda > \bar{\lambda} \equiv (\beta - 2v)/\beta$, and always leads to a P-OSP-OS outcome. DL introduction is profitable for the vendor under a restricted set of conditions (see Appendix 3) and user surplus is always increasing whenever Dual License is possible. Proof: see Appendix 3.

On the one hand, Proposition 7 extends the results of Proposition 6 to the case where the outcome is initially only partially-served. When DL is profitable, the price charged by the vendor
always increases after introduction of DL. As before, DL may or may not be profitable depending on the parameters. While the conditions for DL to be profitable are more complex than those in the previous case (see detailed conditions in Appendix 3), the role played by the spillovers and the degree of compatibility is qualitatively the same. As before, whenever possible, DL always increases the surplus of all users. Again, since the vendor’s profit is not always increasing in the DL case, the same type of “conflicts of interest” may arise between users and the vendor.

However, considering this situation generates new insights about the DL decision. Considering a fully-served market before DL introduction, when profitable, the effects of this introduction was unambiguous from the software vendor’s perspective (business-stealing effect that induces lower P market share and higher price). Proposition 7 reveals that this business-stealing effect may disappear when the market is not fully covered prior DL introduction. Hence, OSP may act as a pure complement to P with no negative influence on the original P software. The price effect is here reinforced by a positive volume effect. This situation may appear as the most favorable one. However, note that it does not exclude that business-stealing effect may benefit to the software vendor also.

5 Managerial and Welfare implications

In the previous sections, we analyzed how the introduction (through dual licensing) of a third software (OSP) which inherits some of the properties of Open Source Software, may alter the competition between traditional Open Source (OS) software and Proprietary Software (P). We highlighted two relevant contexts: one in which DL introductions leads to crowd out OS from the market. Another in which DL leads to shared-outcomes. In both contexts, we analyzed how dual licensing impacts on market shares, the prices and profit of the commercial vendor, and welfare. Our results are largely compatible with those of Comino and Manenti (2011) that illustrate the role of programming spillovers. However, our framework introduces competition and allows for a better understanding of substitution and complementarity effects between products. We also fully analyze the case of a not fully-served market. This analysis allows to better understand how DL may increase market coverage.

DL, OS deterrence and pricing strategy. We first showed that OS deterrence could only be achieved if high development costs on the OS software compared to those incurred by developers using the P software. From a managerial perspective, it implies that even if the software vendor benefit from large spillovers, no OS deterrence can be achieved if the condition on development costs is not met. We also evidenced an interesting pricing strategy based only on the spillovers from the hybrid software to the proprietary software. This case also exhibited the strong substitution effects between the hybrid and the proprietary software where a part of the users of the proprietary software leave this software and switch to the OSP software. Despite these substitution effects, we proved that a DL strategy will be profitable whenever the spillovers coming from the OSP software are sufficiently large and gave a simple and sufficient necessary condition for that. The corollary to this last point is that commercial vendors may sometimes prefer an accommodating strategy to DL. Despite OS software is crowded out, we depart from usual arguments about predatory pricing in two ways. One one hand, predatory pricing often results of a maximization program of a sum of discounted profits vs a one-shot profit. This is not the case here since OS deterrence is the induced effect of a simple profit maximization. Second, predatory pricing generally implies lower price level to deter potential incumbents. When we compare the situation prior and after DL, we find that DL (whenever profitable) leads to higher price thanks to the spillovers. Note that this situation is not directly driven by standard predatory pricing argument.
In terms of the business strategy of the vendor, DL may at first sight be compared to the simultaneous distribution of a “freemium” and “premium” version of a software that corresponds to a special form of second-degree price discrimination. A major difference here is that the two versions may not be implemented separately (by say, an vendor’s investment in software quality). More specifically, the equivalence of the freemium software (here OSP) is a necessary condition for the “premium” strategy to be implemented (here P): the presence of the freemium product generates positive spillovers which increase the utility of users of the “premium” product (P software). Unlike standard models of product differentiation and versioning (Belleflamme (2006)), our framework add two elements. First, in our setting, this product differentiation incurs at no cost for the vendor since the increase in quality is here brought by the freemium users and does not derive from an increase in the vendor’s R&D effort. A second key difference is that the amount of differentiation between the two items cannot be decided strategically by the provider of the two items (here the software vendor) but is here user-driven (as an effect of the activity of the OSP community).

Product substitution, market coverage and welfare. To analyze these issues, we considered a situation where competition is relatively balanced prior to DL introduction (same intrinsic utility brought by the proprietary and OS software, same development costs incurred by users when adopting the two software). In this context, we observe the crucial roles of the degree of compatibility and of the magnitude of spillovers on the DL decision. When DL is introduced on a full-served market, it can only induce some business-stealing effects for the original proprietary software. We also show that DL introduction never leads to a decrease in OS market shares. Hence, the introduction of DL partially divert some users from the P software to the OSP software. Despite this, we showed that this strategy may be profitable. The conditions for that imply both the magnitude of the spillovers and the compatibility degree. Large spillovers provide sufficient conditions for DL to be profitable. If spillovers are not so large, a lower compatibility degree may help to DL. If the market is not fully served before DL introduction, we showed that DL introduction will necessarily lead to full market coverage. In this situation, we showed that OS market share is always higher after DL introduction. The evolution of P market share is more ambiguous. We highlighted a situation in which both P market share and the price of the P software may increase. In this case, the complementarity degree between OSP and P is maximal and the software vendor benefit from OSP by both increasing its user base and its price.

It is interesting to note that users always benefit from DL introduction. Conclusions on welfare are more nuanced. We evidenced conflicts of interests, resulting here in Pareto sub-optimal outcomes. In those cases, while total welfare (including both the profit to the commercial vendor and the surplus of all users) increases, thanks to DL, the profit to the commercial vendor decreases with dual licensing. In those situations, DL is not introduced and this might justify some public intervention to encourage the practice of dual licensing strategies from vendor’s viewpoint. When feasible, this could legitimize some lump-sum transfer to help DL diffusion.

The key role of development spillovers and product compatibility. A general implication of this work is related to the strategy of the proprietary software vendor. In all cases, our results clearly show the prevalence of two key parameters. Parameter $a$ measuring the amount of spillovers derived from the OSP community and brought into the proprietary software, whatever the cases examined below, has a positive effect on the decision of the commercial vendor to DL and on its profit when using DL. Returning to the vendor’s strategy, it is clear that one of its major goal should be to implement strategies that would maximize the value of $a$ whenever possible. These strategies can take at least two different forms. One is related to the management of the OSP software. A well-managed OSP community will more easily provide new features for the proprietary software by enhancing code transfers from the OSP to the P software. This may be achieved by an increased (though costly) involvement of the vendor in the OSP community. Alternatively, Parameter $a$ can also be interpreted as the degree of openness of the OSP license (e.g. by playing on the possibility of
re-use, of further distribution, etc.). When the licensing terms of the OSP software are more “free” (ultimately converging to the GPL license), the vendor may find it difficult to capture the spillovers coming from the OSP software. However, manipulating the degree of openness of the OSP license may be hazardous. On the one hand, when decreasing the degree of openness, the vendor may be more likely to capture the developers’ effort on the OSP software. On the other hand, as suggested by some empirical observations between developers’ incentives and the degree of license restriction (Fershtman and Gandal (2007)), the intensity of the developers effort may decline if developers expect their effort to be captured by the vendor. For that reason, Parameter $a$ cannot be thought of as entirely driven by the vendor and this issue should be analyzed separately.

Finally, our results emphasize the role of the level of compatibility between the OS and the OSP software. This degree of compatibility clearly has a negative impact on the vendor’s profit in the cases examined here: when the compatibility between the OS and the OSP software increases, the substitution between the OSP and the P software increases and the final effect is detrimental to the commercial vendor. This may suggest that, in the cases examined below, the vendor should have some incentive to limit the compatibility between its OSP software and the OS software. However, it should be kept in mind that such a strategy would be conditional on the strategy related to the OS software (since compatibility may be uni- or bi-directional). Hence, acting on the level of compatibility must take account of the potential reactions of the OS community. Further, despite this final effect on profit in the cases examined here, the compatibility may also have positive effects (even for the commercial vendor). For instance, increasing the compatibility between the OSP and the OS software may sometimes could generate a higher OSP market share and benefit the vendor through the spillovers derived from OSP.

6 Conclusion

This paper analyzes the impact of dual licensing (DL) on software vendor’s strategy and welfare in the presence of ex ante competition between an Open Source and a proprietary software platform. It proposes a simple theoretical modeling focusing on adoption issues when dual licensing is introduced in this competitive setting. In this setting, we analyse in depth two relevant cases. In the first case, the DL introduction induces OS crowding out. We show that this strategy is profitable only if development spillovers are sufficiently large. This comes with a reduction of the original proprietary software market share and an increase in price. The second case highlights a situation in which the characteristics of the original open source and proprietary software are rather close. In this setting, we show that complementarities between the dual and the proprietary software may arise. However, we stress the existence of conflicts of interests that lead to an insufficient implementation of DL strategies from a welfare point of view.

The discussion on the managerial implications of our analysis opens up one general issue about the generalization of our results to other types of industries. While our paper is designed for the software industry only, similar patterns (e.g. role of external user-led knowledge and spillovers) may be observed in other industries and give birth to hybrid business models based on “open innovation” (Chesbrough (2006)). Even though dual licensing is very specific to the software industry, this paper emphasizes on the role of standards (here compatibility) and spillovers using hybrid models. Further analysis based on case studies coming from other industries is needed to evaluate the potential of similar business strategies for other industries.
7 References


8 Appendix

8.1 Appendix 1: Computation of the benchmark case

*Proof of Lemma 1*

In this situation, all users initially adopt one software and divide across the P and the OS software. Users located close to 0 adopt the P software while those located close to 1 adopt the OS software. The user (noted \(i_{p/os}\)) indifferent between the two software is defined by \(U_{p}(i_{p/os}) = U_{os}(i_{p/os})\). Hence, \(i_{p/os} = (v + \beta - u - p)/(\alpha + \beta)\). From that, we deduce the expression of the profit \(\pi = p_{p/os}\) and obtain from the FOC that \(p^{*} = (v + \beta - u)/2\) and \(m^{*}_{p} = (v + \beta - u)/2(\alpha + \beta)\). The optimal profit of the vendor is then \(\pi^{*} = (v + \beta - u)^{2}/4(\alpha + \beta)\). Putting together the boundary conditions for \(i^{*}_{p/os}(0 < i^{*}_{p/os} < 1)\), the condition for the utility of all users to be strictly positive (\(U_{p}(i_{p/os})^{*} > 0\)) and the second-order condition, we obtain the following existing conditions for this equilibrium. The condition \(v > u - \beta\) has to be always checked. Besides, either the condition \(2\alpha + \beta < v < u + 2\alpha + \beta\) or the condition \((2\alpha + \beta)(\beta - u)/\beta < v \leq 2\alpha + \beta\) need to apply. In these conditions, we can deduce P users’ surplus \(W^{*}_{p}\) and of OS users’ surplus \(W^{*}_{os}\) from the following definitions: \(W^{*}_{p} = \int_{0}^{i_{p/os}} U_{p}^{*}(i)di\) and \(W^{*}_{os} = \int_{i_{p/os}}^{1} U_{os}(i)di\). We can thus deduce the total surplus of all users \(W^{*}_{users} = W^{*}_{p} + W^{*}_{os}\) and the total welfare \(W^{*} = W^{*}_{users} + \pi^{*}\)

*Proof of Lemma 2*

In this situation, users located close to 0 adopt the P software first, while those located close to 1 adopt the OS software first. Between these two populations, some users obtain a negative utility from the two software types and do not adopt neither software. User \(i_{p/o}\) is defined by a null utility when adopting the P software \(U_{p}(i_{p/o}) = 0\) (and thus \(m_{p} = i_{p/o}\)). In defining this this equilibrium, we impose that \(U_{os}(i_{p/o}) < 0\) (if not, we would be in the previous situation with full adoption). Similarly, User \(i_{o/OS}\) is defined by a null utility when adopting the OS software \(U_{os}(i_{o/OS}) = 0\)
and thus \( m_{\text{OS}} = 1 - i_{\text{OSP}} / i_{\text{OS}} \). Again, by definition of this equilibrium, we impose that \( U_p(i_{\text{OSP}} / i_{\text{OS}}) < 0 \) (if not, we would be in the previous situation with full adoption). These two restrictions imply that 0 < \( i_{P/\text{OSP}} < i_{\text{OSP}/\text{OS}} < 1 \) (boundary conditions).

We deduce that \( i_{P/\text{OSP}} = (v - p/\alpha) \) and that \( i_{\text{OSP}/\text{OS}} = (u - \beta) / \beta \) and thus \( m_{\text{OSP}}^* = u / \beta \). From that, we deduce the profit \( \pi = p i_{P/\text{OSP}} \) and obtain from the FOC that \( p^* = v / 2 \) and \( m_p^* = v / 2 \alpha \). The optimal profit of the vendor is then \( \pi^* = v^2 / 4(\alpha) \). Putting together the boundary conditions and the second-order condition, we obtain the following existing conditions for this equilibrium: \( v < 2 \alpha \) and \( \beta > 2 u \alpha / (2 \alpha - v) \). In these conditions, we can deduce P users’ surplus \( W_p^* \) and of OS users’ surplus \( W_{\text{OSP}}^* \) from the following definitions: \( W_p^* = \int_{i_{P/\text{OSP}}}^0 U_p^*(i)di \) and \( W_{\text{OSP}}^* = \int_{i_{\text{OSP}/\text{OS}}}^1 U_{\text{OSP}}(i)di \). We can thus deduce the total surplus of all users \( W_{\text{users}}^* = W_p^* + W_{\text{OSP}}^* \) and the total welfare \( W^* = W_{\text{users}}^* + \pi^* \).

8.2 Appendix 2: Computation of the DL case

**Proof of Lemma 3**

Suppose that the three software are active.

(a) Suppose then that the agent located at \( i = 0 \) does not adopt the P software. Then, given that whatever \( p \) and \( m_{\text{OSP}} \), \( U_p(i) \) decreases with \( i \) more rapidly than the other utility levels. If so, no other agent adopts the P software, which contradicts our initial assumption. Hence, if the three software products are all active, the agent located at \( i = 0 \) adopts the P software and so do agents located in its neighborhood.

(b) Let now consider the decision of the ‘neighbors’ of P software users. Define by \( i^* \) the last agent adopting the P software and suppose that this agent is indifferent between using the P software and reservation. Given that \( U_{\text{OSP}}(i) \) decreases with \( i \) while the reservation utility does not depend on \( i \), the reservation strategy always dominates adoption of the OSP software for all agents characterized by a location \( i > i^* \). In such a case, there are no OSP adopters. This contradicts our initial assumption. We deduce that there is no agent indifferent between using the P software and reservation.

(c) For the same reasons developed in (a), the agent located at \( i = 1 \) and agents located in its neighborhood adopt the OS software. As \( U_{\text{OS}}(i) \) increases with \( i \) while the other utilities do not, all agents adopting OS are close to \( i = 1 \) and there is no agent indifferent between P and OS when some agents adopt OSP.

Finally, we can deduce that when they are represented, OSP adopters are adjacent to P adopters.

**Proof of Proposition 1**

Given Lemma 3 and given the form of the utility functions associated with the adoption of P, OS and OS, users with the lowest values of \( i \) adopt the P software first, users with intermediate values of \( i \) adopt the OSP software and users with the highest values of \( i \) adopt the OS software. The user indexed \( i_{P/\text{OSP}} \) is indifferent about the P and OSP software is such that \( U_p(i_{P/\text{OSP}}) = U_{\text{OSP}}(i_{P/\text{OSP}}) \).
Hence, \( i_{p/\text{osp}} = (am_{\text{osp}} - p)/\alpha \). Similarly, the user indexed \( i_{\text{osp}/\text{os}} \) is indifferent about the OSP and the OS software and is defined by \( U_{\text{osp}}(i_{\text{osp}/\text{os}}) = U_{\text{os}}(i_{\text{osp}/\text{os}}) \). Hence, \( i_{\text{osp}/\text{os}} = (u - v + \beta(\lambda - 1))/(\alpha + \beta)(\lambda - 1) \). This case is valid only if \( 0 < i_{p/\text{osp}} < i_{\text{osp}/\text{os}} < 1 \) (boundary conditions). The profit then is \( \pi = p \cdot i_{p/\text{osp}} \). From the FOC, we obtain \( p^* = a(u - v + \beta(\lambda - 1))/(\alpha + \beta)(\lambda - 1) \) and \( m^* = a(u - v + \beta(\lambda - 1))/(\alpha + \beta)(\lambda - 1)(a + \alpha \lambda) \). We also deduce the number of OSP and OS users at equilibrium: \( m_{\text{osp}}^* = (u - v + \alpha(\lambda - 1))/(\alpha + \beta)(\lambda - 1) \) and \( m_{\text{os}}^* = (a + 2a\lambda)(u - v + \beta(\lambda - 1))/(\alpha + \beta)(\lambda - 1)(a + \alpha \lambda) \). From that, we can compute the profit which can be expressed as \( \pi^* = a^2(v - u + \beta(1 - \lambda))^2/(4(\alpha + \beta)^2(\lambda - 1)^2(a + \alpha \lambda)) \). Putting together the boundary conditions and the second-order condition, the equilibrium exists if either conditions 1, conditions 2 and conditions 3 (as defined below) apply.

- Conditions 1: \( u < v \) and \( \alpha < (v - u)/(1 - \lambda) \) and either \( \beta > u/(1 - \lambda) \) or \( \beta > u/(1 - \lambda) \) and \( \alpha < v\beta/(\beta(1 - \lambda) - u) \).
- Conditions 2: \( u \geq v \) and either \( u/(1 - \lambda) \leq \beta < (v - u)/(1 - \lambda) \) or \( \beta < u/(1 - \lambda) \) and \( \alpha < v\beta/(\beta(1 - \lambda) - u) \).
- Conditions 3: \( u = v \) and \( u/(1 - \lambda) \leq \beta \).

In these conditions, we can deduce \( P \) users’ surplus \( W^*_P \), of OSP users’ surplus \( W^*_\text{osp} \) of OS users’ surplus \( W^*_\text{os} \) from the following definitions: \( W^*_P = \int_0^{i_P} u(p) \, dP \); \( W^*_\text{osp} = \int_{i_{\text{osp}/\text{os}}}^{1} u(\text{osp}) \, d\text{osp} \); and \( W^*_\text{os} = \int_{i_{\text{osp}/\text{os}}}^{1} u(\text{os}) \, d\text{os} \). We can thus deduce the total surplus of all users \( W^*_\text{users} = W^*_P + W^*_\text{osp} + W^*_\text{os} \) and the total welfare \( W^* = W^*_\text{users} + \pi^* \).}

**Proof of Proposition 2**

Given Lemma 3 and the form of the utility functions associated with the use of \( P \), OS and OS, and considering increasing values of \( i \), users with smaller \( i \) adopt the \( P \) software, then users with intermediate \( i \) adopt the OSP software. Users with highest values of \( i \) adopt the OS software. The user indexed by \( i_{p/\text{osp}} \) is indifferent between the \( P \) and OSP software and is such that \( U_P(i_{p/\text{osp}}) = U_{\text{osp}}(i_{p/\text{osp}}) \). Hence, \( i_{p/\text{osp}} = (am_{\text{osp}} - p)/\alpha \). Similarly, the user indexed by \( i_{\text{osp}/\text{os}} \) is indifferent between the OSP and the OS software and is defined by \( U_{\text{osp}}(i_{\text{osp}/\text{os}}) = U_{\text{os}}(i_{\text{osp}/\text{os}}) \). Hence, \( i_{\text{osp}/\text{os}} = (u - v + \beta(\lambda - 1))/(\alpha + \beta)(\lambda - 1) \). This case is valid only when \( 0 < i_{p/\text{osp}} < i_{\text{osp}/\text{os}} < 1 \) (boundary conditions).

We know from Lemma 3 that the \( P \) and OSP are necessarily adjacent, and from Lemma 2 that non-adopting users are necessarily located between \( P \) and OS users. Then, for values of \( i \) higher than \( i_{p/\text{osp}} \), users adopt the OSP software as long as \( U_{\text{osp}}(i) > 0 \). Consider increasing values of \( i \). The ‘last’ OSP user (indexed \( i_{\text{osp}/\os} \)) is then defined by \( i_{\text{osp}/\os} = 0 \). Hence, \( i_{\text{osp}/\os} = v/\alpha(1 - \lambda) \).

At the opposite side of the segment, users adopt the OS software as long as \( U_{\text{os}}(i) > 0 \). When \( i \) decreases from \( i = 1 \), the ‘last’ user to adopt OS (indexed \( i_{\os}/\os \)) is then defined by \( i_{\os}/\os = 0 \). Thus, \( i_{\os}/\os = 1 - (u/\beta(\lambda - 1)) \). We thus obtain three critical values of \( i \) and this case is valid only when \( 0 < i_{p/\text{osp}} < i_{\text{osp}/\os} < i_{\os}/\os < 1 \) (boundary conditions).

By definition, we thus have, \( m_p = i_{p/\text{osp}} \); \( m_{\text{osp}} = i_{\text{osp}/\os} - i_{p/\text{osp}} \); and \( m_{\os} = 1 - i_{\os}/\os \). From this expression, we compute the profit \( \pi = p \cdot i_{p/\text{osp}} \) and obtain from the FOC that \( p^* = av/2a\alpha(1 - \lambda) \) and \( m^* = av/(2a\alpha(\lambda - 1)(a + \alpha \lambda)) \). We also deduce the number of OSP and OS users at equilibrium: \( m_{\text{osp}} = av/(2a\alpha(\lambda - 1)(a + \alpha \lambda)) \) and \( m_{\os} = u/\beta(1 - \lambda) \). From that, we can compute the profit of the vendor and the surplus for all categories of users. The optimal profit of the vendor is then \( \pi^* = a^2v^2/(4a^2(\lambda - 1)^2(a + \alpha \lambda)) \).
1)^2(a + \alpha \lambda)). Putting together the boundary conditions and the second-order condition, we obtain the following existing conditions for this equilibrium: \( v < \alpha(1 - \lambda), u < \beta(1 - v/\alpha) \) and \( \alpha > \beta \).

Similarly to the proof of Proposition 1, we deduce equilibrium surpluses from their definition:
\[
W_p^* = \int_0^{i_p^{osp}} U_p(i)\,di; \quad W_{osp}^* = \int_{i_0/osp}^1 U_{osp}(i)\,di; \quad \text{and} \quad W_{osp}^* = \int_{i_p/osp}^1 U_{osp}(i)\,di.
\]
We can thus deduce the total surplus of all users \( W_{users}^* = W_p^* + W_{osp}^* + W_{osp}^* \) and the total welfare \( W^* = W_{users}^* + \pi^* \).

**Proof of Proposition 3**

In the situation described by Proposition 3, users adopt either the P software or the OSP software. This case is close to the first situation (P-OSP-OS) in that the market is fully served, but differs in that no user adopts the OS software. Considering increasing values of \( i \), users adopt the P software first and then the OSP software. As previously, the user (noted \( i_p/osp \)) indifferent about the P and OSP software is defined by \( U_p(i_p/osp) = U_{osp}(i_p/osp) \) and is characterized by \( i_p/osp = (am_{osp} - p)/(\alpha \lambda) \). Since all other users adopt the OSP, the user indifferent about the OSP and OS software (noted \( i_{osp/osp} \)) is characterized by \( U_{osp}(i_{osp/osp}) = U_{osp}(i_p/osp) \) and is thus defined by \( i_{osp/osp} = (u - v + \beta(\lambda - 1))/((\alpha + \beta)(\lambda - 1)) \) as above. However, this user is here located outside the unitary segment since adoption of the OS software is never a dominant strategy here. Then, this case is valid only if the following condition holds: \( 0 < i_{p/osp} < 1 < i_{osp/osp} \) (boundary conditions). We also need to check that all users receive a positive utility level when adopting the OSP. A sufficient condition for that is \( U_{osp}(1) > 0 \). This conditions means that the user located at position \( i = 1 \) (i.e. the user that gets the lowest utility level when adopting the OSP) obtains at least a positive utility level.

Profit is expressed as above (\( \pi = p \cdot i_p/osp \)) and we obtain from the FOC that \( p^* = a/2 \) and \( m_p^* = a/(2(\alpha + \alpha \lambda)) \), we also deduce the number of OSP users at equilibrium: \( m_{osp}^* = 1 - m_p^* = 1 - a/2(\alpha + \alpha \lambda) \). Putting together the boundary conditions and the second-order condition, we obtain the following existence conditions for this equilibrium: \( u + \alpha < v + \alpha \lambda \) and \( v + \alpha \lambda > \alpha \).

From that, we can compute the profit of the vendor and the surplus of all categories of users. The optimal profit of the vendor is then \( \pi^* = a^2/(4(\alpha + \alpha \lambda)) \). Similarly to the proof of Proposition 1, we deduce equilibrium surpluses from their definition: \( W_p^* = \int_0^{i_p/osp} U_p(i)\,di \) and \( W_{osp}^* = \int_{i_p/osp}^1 U_{osp}(i)\,di \). We can thus deduce the total surplus of all users \( W_{users}^* = W_p^* + W_{osp}^* \) and the total welfare \( W^* = W_{users}^* + \pi^* \).

**Proof of Lemma 4**

*Lemma 4* There is no equilibrium outcome where only the OSP and the P software are active and some users do not adopt \( (m_{osp} + m_p < 1 \) and \( m_{oss} = 0) \).

*Proof.* From Lemma 3, we can infer that, were such a case to exist, the agent located at \( i = 1 \) would not adopt any software. Then, consider expression 1, and note that if the agent located at \( i \) would have chosen the OS software without the introduction of the OSP software, then the utility of this agent (without DL) would have been \( U_{osp}(1) = u - \beta \). If this utility was previously non-negative, its utility level with DL \( (U_{osp}(1) = u - \beta(1 - \lambda)) \) is greater than its utility level without DL and is strictly positive. Its utility level being positive, this user then is still a potential OS adopter with DL. We conclude from this reasoning that the only case of exclusion of the OS software corresponds to the case of Proposition 3 ■
8.3 Appendix 3. DL decision

\textbf{Proof of Proposition 4}

We demonstrate Proposition 4 by considering two cases (initial full adoption, initial partial adoption separately (Lemmas 5 and 6 respectively) and then merge the results of the two lemmas to derive the proposition.

Let us first consider the case of initial full adoption (\textit{i.e.} P-OS equilibrium as depicted by Lemma 1). Since we are interested here in outcomes where the OS is crowded out after DL is introduced (P-OSP cases as depicted by Proposition 3), the vendor has an incentive to dual license its software only if $\pi^*_p_{-osp} > \pi^*_p_{-os}$ (where $\pi^*_p_{-osp}$ and $\pi^*_p_{-os}$ depict the optimal profit in the P-OSP and P-OS cases respectively).

\textbf{Lemma 5} When the outcome is initially fully served (P-OS outcome) and when the introduction of the DL leads to a P-OSP outcome, then the vendor may or may not have an incentive to dual license on a restricted set of parameters only.

\textit{Proof.} Putting together the set of existence conditions of Lemma 1, that of Proposition 3 and the condition on profit $\pi^*_p_{-osp} > \pi^*_p_{-os}$, the conditions for DL to be profitable in that case are as follows:

\[ (v - u + \beta)^2/(\alpha + \beta) + \sqrt{((-u + v + \beta)^2 + 4(\alpha(v - u + \beta)^2(\alpha + \beta)))/(\alpha + \beta)^2} > 2\alpha \]

and (\((\alpha > 0 \text{ or } \sqrt{2}\beta \leq \alpha \text{ or } \sqrt{6}\beta \leq 5\alpha + \beta) \text{ and } ((u + \alpha = v \text{ and } (v \geq 2\alpha + \beta \text{ or } 2v > 2\alpha + \beta)) \text{ or } (u + \alpha < v \text{ and } (v = 2\alpha + \beta) \text{ or } (v > 2\alpha + \beta \text{ and } u + 2\alpha + \beta > v) \text{ or } (v > 2\alpha + \beta \text{ and } 2v > 2\alpha + \beta \text{ and } u + (v/(2\alpha + \beta) > \beta)))) \text{ or } (u < v \text{ and } 1 + u/\alpha < v/\alpha + \lambda \text{ and } ((u + \alpha > v \text{ and } (2v \geq 2\alpha + \beta \text{ and } (\sqrt{2}\beta = 2\alpha \text{ or } \sqrt{6}\beta = 5\alpha + \beta)) \text{ or } (v \geq 2\alpha + \beta \text{ and } \sqrt{2}\beta \leq 2\alpha \text{ or } 5\alpha + \beta)) \text{ or } (2v > 2\alpha + \beta \text{ and } \sqrt{6}\beta > 5\alpha + \beta) \text{ or } (v < 2\alpha + \beta \text{ and } (\sqrt{2}\beta < 2\alpha \text{ or } (\sqrt{2}\beta > 2\alpha \text{ and } \sqrt{6}\beta < 5\alpha + \beta)))))) \text{ or } (u + (v/(2\alpha + \beta)) > \beta \text{ and } 2v > (\beta(2\alpha + \beta))/(\alpha + \beta) \text{ and } (2v < 2\alpha + \beta \text{ and } (\sqrt{2}\beta = 2\alpha \text{ or } \sqrt{6}\beta = 5\alpha + \beta) \text{ or } (\sqrt{2}\beta < 2\alpha \text{ or } (\alpha > 0 \text{ and } \sqrt{6}\beta > 5\alpha + \beta) \text{ or } (\sqrt{2}\beta > 2\alpha \text{ and } \sqrt{6}\beta < 5\alpha + \beta)))))) \]

When $\lambda = 0$, this set reduces to $u + \alpha < v$ and $\sqrt{(v - u + u + \beta)^2} + (v - u + \beta)^2)/(\alpha + \beta) = \bar{\alpha} < a$ and $((2\alpha + \beta = v \text{ or } (v > 2\alpha + \beta \text{ and } u + 2\alpha + \beta > v) \text{ or } (2v > 2\alpha + \beta \text{ and } u + (v)/(2\alpha + \beta)) > \beta \text{ and } v < 2\alpha + \beta))$.

Conversely, when putting together the set of existence conditions of Lemma 1, that of Proposition 3 and the inverse condition on profit $\pi^*_p_{-osp} < \pi^*_p_{-os}$, one can show that this set is a non-empty subset. Hence, the set of parameters corresponding to an increase in profit is more restrictive than the set defined by Lemma 1, and of Proposition 3 only, and we can infer that DL may sometimes be profitable but not always. $\blacksquare$

Let us now consider the case of partial initial adoption (\textit{i.e.} P-OS equilibrium as depicted by Lemma 2). Since we are interested here in outcomes where the OS is crowded out after the DL is introduced (P-OSP cases as depicted by Proposition 3), the vendor has an incentive to dual license its software only when $\pi^*_p_{-osp} > \pi^*_p_{-os}$ (where $\pi^*_p_{-osp}$ and $\pi^*_p_{-os}$ depict the optimal profit in the P-OSP and P-OS cases respectively). Similarly to the previous situation, we come to Lemma 6 still assuming that $\lambda = 1$ in the long run:
Lemma 6 When the outcome is initially partially covered and when the introduction of the DL lead to a
P-OSP outcome, then the vendor may have an incentive to dual license on a restricted set of parameters
only.

Proof. Putting together the set of existence conditions of Lemma 2, Proposition 3 and the condition
on profit \( \pi^*_p > \pi^*_p - \text{osp} \), the conditions for DL to be profitable in that case are as follows (with
\( \lambda \geq 0 \)):

\[
\begin{align*}
    u + \alpha &< v + \alpha \lambda \text{ and } -(2u)/(v - 2\alpha) < \beta \text{ and } \lambda < 1 \\
\end{align*}
\]

(i) \( \alpha = v \) or (ii) \( v < \alpha \) with \( v/\alpha + \lambda > 1 \) or (iii) \( \alpha < v < 2\alpha \)

When \( \lambda = 0 \), this set reduces to \( \frac{v^2}{\alpha} < a \text{ and } \frac{2u}{v - 2\alpha} < \beta \text{ and } u + \alpha < v \text{ and } v < 2\alpha \text{ and } v > \alpha \)

Conversely, putting together the set of existence conditions of Lemma 2, that of Proposition
3 and the inverse condition on profit \( \pi^*_p - \text{osp} < \pi^*_p - \text{os} \), we can show that this set is generally a
non-empty set. Hence, the set of parameters corresponding to an increase in profit is more restric-
tive than the set defined by Lemma 2, and of Proposition 3 only, and we can infer that DL may
sometimes be profitable but not always.

Putting Lemmas 5 and 6 together, we can check that whatever the initial outcome (fully or
partially covered market), the DL strategy may be profitable for a restricted set of parameters.
Considering the values of \( a \) for that, one sufficient condition is that \( a > \max \tilde{a}, \frac{v^2}{\alpha} \).

Proof of Proposition 5

Similar to Proposition 4, we demonstrate Proposition 5 by considering two cases (initial full adop-
tion, initial partial adoption) separately (Lemmas 7 and 8 respectively) and then merge the results
of the two lemmas to derive the proposition.

Lemma 7 When the market is initially fully served (P-OS outcome), when DL leads to OS crowding out
(O-OSP outcome) and when introducing DL is profitable for the vendor (\( \pi^*_p - \text{osp} < \pi^*_p - \text{os} \)), DL always
decreases the market share of the P software and increases the amount of the fees paid by P users.

Proof: This is based on comparison of the optimal market share of the proprietary software \( m^*_p \)
under the two outcomes depicted by Lemma 1 and 3. Under the existence conditions of the two out-
comes and the profitability condition (\( \pi^*_p - \text{osp} > \pi^*_p - \text{os} \)), we can show that \( m^*_p \text{[P-OSP]} < m^*_p \text{[P-OS]} \)
ever holds (where \( m^*_p \text{[P-OS]} \) and \( m^*_p \text{[P-OSP]} \) denote the optimal P market share in the P-OS
and P-OSP outcomes respectively). Then the market share of the P software always increases when
switching to DL. The same reasoning applies to the optimal price with and without DL.

Lemma 8 When the market is not fully served (P-OS outcome), when DL leads to OS crowding out
(O-OSP outcome) and when introducing DL is profitable for the vendor (\( \pi^*_p - \text{osp} < \pi^*_p - \text{os} \)), DL always
decreases the market share of the P software in the long run (\( \lambda = 0 \)) and increases the amount of the
fees paid by P users in the short run (\( \lambda \geq 0 \)) and in the long run (\( \lambda = 0 \)).

Proof: This is based on comparison of the optimal market share of the proprietary software \( m^*_p \)
under the two outcomes depicted by Lemma 2 and 3. Under the existence conditions of the two outcomes
The results for the existence conditions are derived from the conditions described by Lemma 1 and Proposition 1 with $u = v$ and $\alpha = \beta$. When comparing the existence conditions of the P-OS outcome to those of other outcomes (P-OSP, P-OSP-\(\emptyset\)-P, as described by Propositions 2 and 3

\[ \text{Proof of Proposition 6} \]

Putting together the existence conditions of the two outcomes and the profitability condition (\(\pi_p^{P-\emptyset-\text{OSP}} > \pi_p^{P-\emptyset-\text{OSP}}\)), we can show that \(m_p^{P-\emptyset-\text{OSP}} < m_p^{P-\text{OSP}}\) never holds (where \(m_p^{P-\emptyset-\text{OSP}}\) and \(m_p^{P-\text{OSP}}\) denote the optimal P market share in the P-\(\emptyset\)-OS and P-OSP outcomes respectively) when \(\lambda = 0\). Yet, in the short run (\(\lambda \geq 0\)), the optimal P market share increases or decreases depending on parameters (i.e. under the existence conditions of the two outcomes with and without DL and the DL profitability condition, we observe either \(m_p^{P-\emptyset-\text{OSP}} < m_p^{P-\text{OSP}}\) or \(m_p^{P-\emptyset-\text{OSP}} > m_p^{P-\text{OSP}}\) depending on the parameters.

The same reasoning applies to the optimal price with and without DL. Under the existence conditions of the two outcomes and the profitability condition (\(\pi_p^{P-\emptyset-\text{OSP}} > \pi_p^{P-\emptyset-\text{OSP}}\)), we can show that \(p_p^{P-\emptyset-\text{OSP}} < p_p^{P-\text{OSP}}\) never holds (where \(p_p^{P-\emptyset-\text{OSP}}\) and \(p_p^{P-\text{OSP}}\) denote the optimal price of the P software in the P-\(\emptyset\)-OS and P-OSP outcomes respectively) whatever the value of \(\lambda\) (0 \(\leq \lambda < 1\)).

Putting Lemma 7 and 8 together, we can check that whatever the initial outcome (fully or partially served outcome) and once the DL strategy becomes profitable, the market share of the P software and the price charged with DL are less than without DL.

\[ \text{DL decision with a balanced outcome and an ex ante non fully covered market} \]

Putting together the existence conditions of the two outcomes and the profitability conditions, the conditions for DL to be profitable can be expressed as follows (where one of Conditions a,b,...) has to be filled

- Condition (a): \(\beta > 2\) and \(\hat{\lambda} < \lambda\) and either \(a \geq \frac{4+2\sqrt{3}}{3} v\) or \(v (1 + \sqrt{5}) < a < \frac{4+2\sqrt{3}}{3} v\)
- Condition (b): \(\beta > \hat{\beta}, a < \frac{8v}{\sqrt{3}}, 2v < a, \text{and } \hat{\lambda} < \lambda\)
- Condition (c): \(a \geq \frac{4+2\sqrt{3}}{3}, \frac{3v}{2} < \beta, \text{and } \beta < 2v\)
- Condition (d): \(a < v (1 + \sqrt{5}), \frac{3v}{2} < \beta, \frac{8v}{\sqrt{3}} < a, \lambda < \hat{\lambda}, \text{and } \beta < 2v\)
- Condition (e): \(a < \frac{4+2\sqrt{3}}{3}, \frac{3v}{2} < \beta, v (1 + \sqrt{5}) < a, \beta < \hat{\beta}, \text{and } \lambda < \hat{\lambda}\)
- Condition (f): \(a < \frac{4+2\sqrt{3}}{4}, v (1 + \sqrt{5}) < a, \hat{\beta} \leq \beta, \text{and } \beta < 2v\)
- Condition (g): \(a < \frac{8v}{\sqrt{3}}, 2v < a, 2v < \beta, \lambda < \hat{\lambda}, \text{and } \beta \leq \hat{\beta}\)
- Condition (h): \(a < v (1 + \sqrt{5}), 2v < \beta, \frac{8v}{\sqrt{3}} < a, \beta < \hat{\beta}, \text{and } \lambda < \hat{\lambda}\)

with \(\hat{\beta} = \frac{4a^2}{u-4uv}, \hat{\lambda} = -\frac{4a^2+2uv}{4uv\beta}\) and \(\hat{\lambda} = -\frac{2\beta+\beta}{\beta}\.}

\[ \text{Proof of Proposition 6} \]
respectively), it is possible to check that these conditions are mutually exclusive. The other results are derived from comparing the different prices, profits and welfare as described by Lemma 1 and Proposition 1 once $u = v$ and $\alpha = \beta$.


Proof of Proposition 7

The results for the existence conditions are derived from the conditions described by Lemma 2 and Proposition 1 when replacing by $u = v$ and $\alpha = \beta$. Comparing the existence conditions of the P-OSP-OS outcome to those of other outcomes (P-OSP, P-OSP-OS, as described by Propositions 1 and 3 respectively), we can check that these conditions are mutually exclusive. The other results are derived by comparing the different optimal prices, profits and welfare as described by Lemma 2 and Proposition 1 once $u = v$ and $\alpha = \beta$.