

## Open source and competition

---

The software market has benefitted for a long time from intellectual property protection. By forbidding free copying and distribution of the proprietary software, the intellectual property protection assured that a proprietary software producer received sufficient revenues to recover its investments in innovation. It therefore seemed necessary to stimulate the software innovation in the first place. Nevertheless, several developers decided to distribute open-source software for free and started using special types of licenses permitting free copying of their products.

The simultaneous presence of the open-source software distributed for free and the proprietary software generating revenues on its sale raises two basic questions. Which type of software will survive? Will software producers continue to invest in innovation at the same pace?

These two simple questions may not have easy answers. The present paper provides an overview of three different opinions presented by Gaudel (2008), Economides & Katsamakas (2005) and Lambardi (2009).

Gaudel (2008) argues that the proprietary software and the open-source software may coexist, as the more general open-source software may be needed to fill in the niches between highly specialized and expensive proprietary software.<sup>1</sup> In other words, the open-source software may serve more general purposes, while the proprietary software may specialize in very narrow areas.

Economides and Katsamakas (2005) demonstrate that the proprietary operating system may dominate the market, because the applications developed for the proprietary operating systems may not have good alternatives working on the open-source operating system and because consumers may not be willing to bear the cost of switching from the proprietary operating system to the open-source operating system.<sup>2</sup>

---

<sup>1</sup> Gaudel A. (2008). *Consumer welfare and market structure in a model of competition between open source and proprietary software*, International Journal of Open Source Software and Processes, 1(2), pp. 43-65.

<sup>2</sup> Economides, N. and E. Katsamakas (2005). *Linux vs. Windows: A comparison of application and platform innovation incentives for open source and proprietary software platforms*, Law and Economics Research Paper Series Working Paper No. 05-20, New York University School of Law.

Finally, Lambardi (2009) indicates that the potential threat of the open-source software may have an adverse effect on proprietary software research and development projects.<sup>3</sup> The possibility of revealing the source code may as a result hinder innovation in the software market.

### Open-source software vs. proprietary software

Software enables human communication with the machine. When writing a software program, developers specify a set of actions to be performed by a computer in a programming language that they understand well (such as Java, C or C++). They later use specially designed computer programs (interpreters and/or compilers) to translate the created source code into the machine code (a sequence of zeros and ones) that may be executed by a machine, but is not well understood by human beings.

Since the machine code is difficult to interpret by human beings, software producers may protect their profits by providing only the machine code and not revealing the source code. This way, a software producer may assure that the developed software would not be easily modified and/or distributed by its competitor.

The software products are protected by copyrights. Copyright is the set of exclusive rights granted to the author or creator of an original work, including the right to copy, distribute and adapt the work. It usually arises with the creation of a work. In other words, for goods that reflect certain degree of creativity, the sole fact that it has been created may be sufficient for the copyrights to apply. As computer programs fall into this category, once created, they automatically benefit from the copyright protections.

The software producers may prefer to conceal the source code rather than to simply rely on the copyright protection. The intellectual property rights do not allow using unauthorized copies of the software, but do not protect from a situation in which a product is sufficiently modified and distributed as a different product. The producers of the proprietary software may hence wish to apply additional measures to protect their profits. Keeping the source code secret is one such a measure.

---

<sup>3</sup> Lambardi, G. D. (2009). *Software Innovation and the Open Source threat*, GREMAQ, Toulouse School of Economics working paper.

Not all the software developers wish to conceal the source code. On the contrary, several software producers create the open-source software revealing both the machine code and the source code. By fully disclosing the source code they promote the free distribution of their software and the modifications of the software by different developers.

The sole disclosure of the source code may be insufficient to implement the free distribution of the open-source software. Facing the automatic protection of the copyrights, developers of the open-source software need to use special types of licenses allowing free copying of their products. For example, open-source software is often distributed under the "GNU General Public License" (hereafter GPL). GPL assures that a distributor of open-source software discloses the source code to all recipients of the software. Furthermore, all the software that contains or is derived from the computer program holding GPL must also hold a GPL.

### Coexistence of the open-source software and the proprietary software

Nowadays, the proprietary software and open-source software co-exist. The interesting question is whether this outcome is sustainable. Gaudel (2008) claims that the two types of software may continue to co-exist and that they may serve different purposes. In particular, the open-source software may serve more general purposes, while the proprietary software may focus on very specific needs.

Gaudel studies a situation in which consumers have different needs with respect to the software and may either contribute to the creation of the open-source software or buy the proprietary software. The proprietary software is produced by firms and sold at a positive price to the consumers. The open-source software involves users in the process of its creation, but is distributed for free. Each user of the open-source software is assumed to devote the same amount of time and effort to develop the open source. As a result, the individual contribution to the creation of the open-source software declines in the number of contributors.

The author shows that the heterogeneity of consumer demand stimulates the creation of different types of software. As a result, large open-source projects may coexist with more specialized proprietary projects. The specialized proprietary software attracts fewer consumers, but better meets their needs. Its consumers are ready to pay for it, because it satisfies their specific needs. The open-source software attracts those consumers whose needs are not well satisfied by the proprietary software. Given that their specific needs are far away from the solutions proposed by the proprietary software, those consumers are not willing to pay for the

proprietary software but prefer to use more general open-source software. What's more, they are not discouraged by the fact that the development of the open-source may require their personal contributions, as being parts of large open-source communities they find their personal involvements little time consuming.

Gaudel does not only show that the two types of software may coexist, but also indicates that their simultaneous presence may be good for the economy. The reason of this finding lies in the observation that the proprietary software producers prefer to serve fewer consumers, meet their specific needs and charge higher prices. In the absence of the open-source software, the market should be thus composed of many expensive highly specialized computer programs. This might not be good for the social welfare, because the prices could remain too high. From the social point of view, it might be better when there are fewer producers of more general software. Since the open-source software developers are willing to offer such general purpose software, their presence could force proprietary software producers to spend less on the development of their products. As a result, end users could benefit from cheaper (yet sometimes less specialized) computer programs.

All in all, Gaudel shows that the co-existence of the open-source software and the proprietary software is possible and may be beneficial for the society.

### Domination of the proprietary software

Economides and Katsamakas (2005) also study the issue of the co-existence of the open-source software and the proprietary software. They focus on the question of possible survival and market expansion of the freshly launched open-source operating system in the presence of the incumbent proprietary operating system. They find that the proprietary operating system may dominate the market leaving a very narrow market segment for the open-source operating system.

Economides and Katsamakas study a specific situation with one proprietary operating system, one open-source operating system and two applications. One application is compatible with the proprietary operating system, while the other is compatible with the open-source operating system. The two applications are not identical, but to a certain degree similar. Certain groups of end users prefer the application compatible with the open-source operating system, while the others have a preference for the application compatible with the proprietary operating system. Each type of users may be willing to switch to the less favorable application if it is offered at an attractive price. Given that the proprietary operating system has been in use before the arrival of the open-source operating system, the end users are already used to it, while need to bear certain switching

cost in order to start using the open-source operating system.

The authors also suppose that each application is offered at the price fixed by its producer. The proprietary operating system is offered at a positive price fixed by the proprietary operating system producer. The open-source operating system is distributed for free, but customers need to devote certain time and effort to learn how to use it.

In this setting, the authors analyze two different specifications. In the first specification, the two applications are produced and distributed independently from the operating systems. In the second specification, the application compatible with the proprietary operating system is produced and distributed by the producer of the proprietary operating system.

The authors show that the proprietary operating system may dominate the open-source operating system in both specifications. In the first specification, three factors contribute to the market domination of the proprietary system. First, the domination of the proprietary operating system is more likely when there are more consumers preferring the application based on the proprietary operating system than the application based on the open-source operating system. This result is rather intuitive: the more consumers like a given good, the easier it is for its producer to dominate the market.

Second, the likelihood of the proprietary operating system domination in the first specification depends on the switching cost from the proprietary operating system to the open-source operating system. The authors suppose that the proprietary operating system is somewhat familiar to the end users, while the decision to start using the open-source operating system is linked with certain switching cost. This switching cost has an immediate effect on the market domination of the proprietary operating system. The more costly it is for the consumers to switch to the open-source operating system, the more likely the proprietary operating system will dominate the market.

Third, the market domination of the proprietary operating system in the first specification becomes easier when the application compatible with the proprietary operating system is not a good substitute for the application compatible with the open-source operating system. In such a situation, the end users preferring the application compatible with the proprietary operating system rarely consider a switch to the other available application even if it is offered at a very attractive price. The producer of the proprietary operating system is then assured that certain consumers will never switch to the open-source operating system and finds it easier to dominate the market.

In the second specification, the sole fact that more consumers are interested in buying the application based on the proprietary operating system is sufficient to assure the market domination of the proprietary operating system. It is hence easier for the producer of the proprietary operating system to dominate the market when it is integrated with the producer of the compatible application. This is because in the presence of the vertical integration the producer of the proprietary operating system is better equipped to compete against the producer of the open-source. In the absence of the vertical integration, both the producer of the proprietary operating system and the producer of the compatible application wish to make profits. What's more, they only consider the effects of their pricing policies on their respective profits and are not concerned by the joint profit maximization. As a result, the price paid for a package of the proprietary operating system and the compatible application is too high and it is more difficult for the producer of the proprietary software to dominate the market. In the presence of the vertical integration, the goal of the integrated entity is to maximize the joint profit generated on the sales of the proprietary operating system and the sales of the compatible application. The price for the package of the proprietary operating system and the compatible application then declines. After such a price cut, it is much easier for the proprietary operating system to dominate the market.

Basing on the model of Economides and Katsamakos one may conclude that market domination of the proprietary software is quite likely. Yet, the development of the open-source software may be stimulated. In the absence of the vertical integration between the producer of the proprietary operating system and the producer of the compatible application, a concerned authority may aim at decreasing the cost of switching from the proprietary operating system to the open-source operating system and at encouraging the application producers to develop applications based on the open-source operating system. In the presence of vertical integration between the application producer and the producer of the proprietary operating system, the market intervention becomes more difficult and the domination of the proprietary operating system may be inevitable.

### Effects of the open-source software on innovation

Lambardi (2009) studies the effect of the presence of the open-source software on the dynamic competition in the software market. In particular, he investigates the following question: how the fact that a proprietary-software producer may become an open-source producer affects the software innovation? In order to address this question, he defines a model with two software producers, each producing basic software and complementary software. The complementary software

produced by one producer is not compatible with the basic software produced by the other producer. The basic software of one of the producer is more innovative than the basic software offered by the other producers. The producer of the more advanced basic software never reveals the source code. The producer of the less advanced basic software may decide to reveal its source code to the public. Both producers may invest in innovation. The return to the investment in innovation depends on whether a producer produces open-source or closed-source software. If a producer opts for closed-source software, it has to bear all the cost of innovation. If however it chooses open-source software, it benefits from help from a community of open-source developers. Given the openness of its source code, the producer of the open-source basic software may not generate any profits on its basic software. It may however generate profits on the complementary software. The producer of the proprietary software generates profits on both basic software and complementary software.

Lambardi shows that the possibility that the developer of the proprietary software may change its business model and start distributing its software as open-source software may have a negative impact on innovation. He argues that the producer of the lagging basic proprietary software has incentives to reveal the source code when the technological lag between the advanced basic software and the lagging basic software is behind certain threshold. The technological gap depends on the investments of the producer of the leading basic software. By decreasing its investments in innovation the producer of the leading basic software may assure that the producer of the lagging basic software will not find it profitable to open its source code. Given that it may be less attractive to compete against freely-distributed open source, the producer of the leading basic software may thus limit its investments to avoid a switch of the competing producer to open source.

Lambardi argues that the subsidy encouraging the producer of the lagging basic software to open its source code may have adverse effects on innovation. He explains that increasing the incentives of the producer of the lagging basic software to open its source may translate into increasing incentives of the producer of the leading basic software to invest less in innovation. As a result of a subsidy supposedly fostering a switch to open source there may in fact be no switch and less innovation.

Lambardi maintains that designing a proper subsidy fostering a switch to open source may be possible albeit difficult. Such a subsidy should negatively depend on the value of the initial technological gap between the available basic computer programs and the value of development help provided by a community of developers of open-source software. In the same time, it should positively depend on the size of population of end users that need only the basic software and would

not use complementary software. Lambardi indicates that the policymaker may not have accurate estimates of these parameters and could hence experience difficulties when designing the subsidy.

Overall, the paper of Lambardi shows that the possibility of revealing the source code may have adverse effects on innovation, because the producer of the leading software may decide to invest less in innovation in order to avoid the switch of the lagging proprietary software into the open source software. In his model, there may be in fact no open-source software available in the market, but the sole possibility of revealing the source code may be sufficient to slow the innovation.

## Conclusions

To conclude, this short article has overviewed three different opinions concerning the impact of the arrival of the open-source software on the future competition in the software market. It first showed that the open-source software applications may co-exist with the proprietary software applications if they serve more general purposes than the proprietary software applications. It then proceeded to discuss difficulties that an open-source operating system may encounter in its market expansion such as reluctance of end-users to switch to the open-source operating system, given that they are already used to the existing proprietary operating system, and the lack of attractive applications developed for the open-source operating system. Finally, it showed that even if there is no open-source software in the market, the sole fact that an existing proprietary application may become open-source software may be sufficient to prevent the producer of the competing proprietary software from investing in projects leading to potential improvements of its software.

Marta Stryszowska

©Microeconomix June 2010

**Microeconomix**  
Economic Analysis Applied to Law

Competition Law & Antitrust  
Regulation  
Energy Economics  
Econometrics

[www.microeconomix.com](http://www.microeconomix.com)