Contents lists available at SciVerse ScienceDirect

Journal of Economic Dynamics & Control

journal homepage: www.elsevier.com/locate/jedc

When to make proprietary software open source ☆

Jonathan P. Caulkins [a], Gustav Feichtinger [b,c], Dieter Grass [b], Richard F. Hartl [d], Peter M. Kort [e,f,*], Andrea Seidl [b,c]

[a] *Carnegie Mellon University, H. John Heinz III College, Pittsburgh, PA, USA*
[b] *Department for Operations Research and Control Systems, Institute for Mathematical Methods in Economics, Vienna University of Technology, Vienna, Austria*
[c] *Wittgenstein Center for Demography and Global Human Capital (IIASA,VID/OEAW,WU), Vienna Institute of Demography/Austrian Academy of Sciences, Vienna, Austria*
[d] *Department of Business Administration, University of Vienna, Vienna, Austria*
[e] *Department of Econometrics and Operations Research & CentER, Tilburg University, Tilburg, The Netherlands*
[f] *Department of Economics, University of Antwerp, Antwerp, Belgium*

A R T I C L E I N F O

Article history:
Received 11 November 2011
Received in revised form
18 December 2012
Accepted 13 February 2013
Available online 1 March 2013

JEL classification:
C61
O32

Keywords:
Open source
Optimal control
Multi-stage modeling
Complementary product
Software

A B S T R A C T

Software can be distributed closed source (proprietary) or open source (developed collaboratively). While a firm cannot sell open source software, and so loses potential sales revenue, the open source software development process can have a substantial positive impact on the quality of a software, its diffusion, and, consequently, the demand for a complementary product from which the firm does profit. Previous papers have considered the firm's option to release software under a closed or open source license as a simple once and for all binary choice. We extend this research by allowing for the possibility of keeping software proprietary for some optimally determined finite time period before making it open source. Furthermore, we study the impact of switching costs.

We show that in case of high in-house R&D costs, the firm always makes the software open source at some point (unless switching itself is too expensive). The timing, however, depends on the initial software quality. On the other hand, when R&D is inexpensive, the firm opens the source code only when the initial level of software quality is low. For intermediate R&D costs, the firm might have the choice between opening the code immediately, opening it at some subsequent and optimally determined time, or keeping it closed forever. Finally we find that whereas high switching costs might prevent firms from adopting an open source business model, low switching costs mainly affect the timing of the opening of the source code.

© 2013 Elsevier B.V. All rights reserved.

1. Introduction

Firms generating revenue by distributing software face some interesting choices that do not arise for more conventional, physical products. One is whether or not to make some of its software "open source" in order to tap the talents of volunteer software developers outside the firm in exchange for giving that software away free. Forgoing revenue by giving a product

☆ This research was supported by the Austrian Science Fund (FWF) under Grants P21410-G16 and P23084-N13.
* Corresponding author at: Department of Econometrics and Operations Research & CentER, Tilburg University, P.O. Box 90153, 5000 LE Tilburg, The Netherlands. Tel.: +31 13 4662062; fax: +31 13 4663280.
E-mail address: kort@uvt.nl (P.M. Kort).

0165-1889/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.jedc.2013.02.009

away seems odd, but can make sense if the firm profits from selling complementary products or services whose demand would increase if the software were adopted more widely.

Haruvy et al. (2008b) examine how a firm should dynamically optimize both prices and investment in software development over time under both open and closed source conditions. Based on this analysis the firm chooses between these two business models and treats that decision as a once and for all binary choice. Here, we generalize Haruvy et al. (2008b) to allow for the possibility of keeping a product closed source for a finite time before than converting it to open source. The analysis assumes that software quality is the crucial state variable and identifies interesting threshold behavior depending on the initial quality. Quality in this context should be understood more generally than merely that the software is free of errors; it also encompasses the existence of features and functionality.

The decision to make a software product open source affects a firm's software development process, its organizational routines, and overall business model. When the source code is freely available, a firm will effectively lose revenues from sales of that software. However, the openness of the code can also have advantages for a firm. As von Hippel and von Krogh (2003) point out, innovation in open source development is typically driven by the users, allowing a software developer to benefit from user contributions with respect to quality, including creativity and security, and, consequently, allowing the firm potentially to reduce in-house software development costs (see also Raymond, 2001; Lerner and Tirole, 2005a).

An open source product also has enormous appeal for customers due to the lack of licensing fees, the possibility to customize the software to their own needs and also to avoid vendor lock-in; see Lerner and Tirole (2002). Thus, making software open will in general increase demand.

Since open source software is mostly given away for free, monetary incentives for a commercial firm to embrace an open source business model come from exploiting complementarities with one of its commercial products. These complementary products can be hard- or software, of course, or consulting services, maintenance contracts, training, certifications etc. See, e.g., Lerner and Tirole (2002) for a consideration of such business models, but also Economides and Katsamakas (2006), Haruvy et al. (2008b), and Kort and Zaccour (2011). Examples of firms following such a strategy include Oracle offering training and consulting services for MySQL and Java, and Google selling apps and mobile ads with its (open source) mobile phone operating system Android.

Haruvy et al. (2003, 2008b) compare the open source and closed source approach in an optimal control framework. They explore conditions under which it is optimal to release software in open vs. closed source when the firm can make profits by the sales of a complementary product, where switching from closed to open source is not allowed. However, quite a few programs, which are today prominent examples of open source software, were initially released under a proprietary license (e.g., Java, MySQL) or are based to a certain extent on closed source software (e.g., Linux and OpenOffice).

Let us take for example a closer look at the programming language Java,[1] developed by Sun and, since its acquisition, by Oracle. Initially released closed source in 1995, it soon became very popular among developers. In order to stimulate the further adoption of Java particularly with respect to its inclusion in Linux distributions,[2] Sun released Java in 2006 under an open source license. The corresponding business model by Sun was to make profits by offering support, training and maintenance, and by dual licensing.

When a firm decides to adapt its existing business model, it first has to clarify several legal issues such as which open source license to use and how to treat users' contributions. From a technical perspective, the firm has to ensure that the source code does not contain, and has no dependencies on, any source code that is not publicly available. Furthermore, the firm has to ensure the availability of necessary infrastructure (such as a code repository system, a website, mailing lists, etc.) and resources (including developers and maintainers), and it has to establish an efficient project governance. For more details on steps that a firm should take when opening source code, see Haddad and Warner (2012). Such actions can be expensive; thus, it makes sense to study the impact of switching costs for adapting the business model, see also Bonaccorsi et al. (2006).

Licensing is a particularly interesting issue. The basic intention of an open source license is to give the user the right to freely use, access, modify and redistribute the source code (The Open Source Initiative, 2012). One can distinguish between licenses with and without copyleft terms. "With copyleft terms" essentially says that if one re-uses the source code, the derivative product must be operated under the same license terms (see, e.g., de Laat, 2005; Lerner and Tirole, 2005a; Bonaccorsi and Rossi, 2003; see also Mustonen, 2003; Lerner and Tirole, 2005b). The purpose of such restrictive terms is to protect the licensor against the commercial usage of their product by competitors and to give contributing users some kind of assurance that their often un-paid work is not directly commercially exploited. Effectively, this also means that once a software is made open source under a restrictive license, it is not possible to make the software closed source again. (Note, however, that some projects, like Oracle's OpenOffice, only accept user contributions if users agree to share or transfer their copyright with the firm, so copyleft restrictions can sometimes be circumvented.)

Even if making an open source software closed source or restricting access to the product is legally possible, such a decision can still risk adverse effects for a firm, such as loss of reputation, loss of (key) contributors, the formation of so-called forks (the open source code is used for a similar, open project by someone else), etc. A recent example is the open

[1] See http://oracle.com.edgesuite.net/timeline/java/ (last accessed February 14, 2013) for the history of Java.

[2] See http://news.cnet.com/Sun-picks-GPL-license-for-Java-code/2100-7344_3-6134584.html (last accessed February 14, 2013).

Download English Version:

https://daneshyari.com/en/article/5098695

Download Persian Version:

https://daneshyari.com/article/5098695

Daneshyari.com