AN ECONOMIC MODEL OF OPEN SOURCE SOFTWARE ADOPTION

By Si Chen

Open Source Strategies, Inc.
Los Angeles, CA, USA

The author is the Principal of Open Source Strategies, Inc., a strategic and technology consulting firm specializing in open source software. He is the project manager of the opentaps Open Source ERP + CRM project and a committer of the Apache OFBIZ project. He has been previously published in *The Journal of Portfolio Management.*

ABSTRACT

This paper examines the competitive strengths of open source versus commercial software from an economic perspective. It uses a simple model of rational software selection to derive the competitive strengths of open source and commercial software models, how they can balance each other, and what the threshold or tipping point for mass adoption of open source software would be. It shows that open source software has the advantage when their commercial competitors have excessively high profit margins or when the software product's features are not scalable due to heterogeneous user requirements.

BACKGROUND

Most software is developed by commercial organizations and sold to end users by granting them the right or license to use the software in exchange for a fee. An alternative model, known as "open source," freely allows users to use, modify, and even distribute the software they use without paying for such a right.

In recent years, open source software has become increasingly popular. The Linux operating system, which began as an open source alternative to Unix in 1991, has emerged as the most visible example of open source software. Another well-known open source product, the Apache web server, is now the most popular application used to distribute content over the Internet, with a market share of over 60%[1]. Other major open source products such as PHP, MySQL, PostgreSQL, and JBoss have also emerged as popular alternatives to commercial software.

The prominence of open source software has led to significant research interest as well. Most of the existing research have focused on the motivations for developing free software (Axelrod (1997), Hertel, Niedner, and Herrmann (2003), Lakhani and Wolf (2005), Lerner and Tirole (2002) and (2005)), the process of open source software development (Mockus, Fielding and Herbsleb (2002), Sacchi (2004), Spinellis and Clemens (2004), Norris and Kamp (2004)), and the use of open source software (Madanmohan (2004), Ortiz (2001), Weber (2004).) Finally, some researchers have used the open source phenomenon to examine more general topics in management and economics. For example, Casadesus-Masanell and Ghemawat (2006) used the

[1] http://www.serverwatch.com/stats/article.php/3653241

competition between Linux and Windows as an example to examine the competitive dynamics of a duopoly.

In this analysis, we will begin with an economic model of why users choose one software product versus another. Then, we will use this model to identify the relative strengths and weaknesses of open source and commercial software models and how they balance each other. Finally, we will use this model to examine the key factors which affect the mass adoption, or "tipping point," of open source and commercial software.

SOFTWARE AS AN ECONOMIC DECISION

For this analysis, we will assume that there is a commercial and an open source software product for a particular need. The commercial software product can be purchased from a software vendor by paying a licensing fee. In addition, the user would pay for services such as setup, configuration, customization, training, and support. The open source product could be used without paying a licensing fee, but the user must pay for additional features as well as any services that might be required to use the software.

A user's decision to use one product versus another is assumed to be a purely economic decision with perfect access to information. In other words, software users are assumed to be fully aware of their required feature set and the true cost of using either a commercial or an open source product. In addition, users are assumed to be rationally value-maximizing so that they would always choose the product which meets their requirements at the lowest cost.

Thus, if K_v is the total cost of using a software product from a commercial vendor and K_{os} is the cost of using an open source software product, then the user is assumed to choose the open source product if

$K_v > K_{os}$

Alternatively, the user is assumed to choose the commercial product if

$K_v < K_{os}$

Finally, we assume that the costs to develop software are the same for both commercial or open source software.

THE TRUE COST OF SOFTWARE

If the true requirements of the users were known and can be expressed as a finite set of features[2] $N = 0 \ldots i$, then we can express the development cost of features required by a user as D_u:

$D_u = \sum D_i$, for $i = 0 \ldots N$,

where D_i is the cost to develop a particular feature for the user.

When a user considers using an open source product, he does not need to pay for a license to use its existing features. He only needs to develop the features which are not already present in the product. Thus, his costs are:

$K_{os} = \sum D_i$, for i over $O = m \ldots n$,

where O is a finite set of features required by the user but not currently available in the open source product. If O is a subset of N ($O \subset N$), then the cost of using open source software can alternatively be expressed as:

$$K_{os} = (1 - P_{os}) * D_u, \qquad (1)$$

where P_{os} is the proportion of user required features which are present in the open source software products, or O / N.

[2] In considering the total cost of software, "features" should include both functions performed by the software and related efforts to make the software usable, such as setup, configuration, customization, training, and support.

In contrast, the user of a commercial software product must pay a license to the vendor of the software to use it. Often, a commercial software product may not support all of a user's needs out of the box, and additional work is required to implement the commercial product after it is purchased from a vendor. Thus, the cost to use a commercial software product can be expressed as:

$K_v = L + I$,

where L is the cost of the license to use the software, I is the additional cost of implementation, and K_v is the total cost to use the commercial software product.

The cost of commercial software licenses is itself a function of the vendor's cost to develop the software plus the vendor's profit margin, divided across the users of the software:

$L = (D_v * M) / (N * S_v)$,

where D_v is the total development costs for feature set V, which are included in the standard software package sold by the vendor, M is the profit margin of the vendor, N is the number of potential users, S_v is the percentage of potential users using and paying for this particular vendor's product.

The cost of implementing the commercial software product is the cost of the user's requirements which are not in the vendor's standard package. These may include additional features special to a particular user ("customizations"), configuration and setup, user training, and ongoing support. This cost can simply be expressed as

$I = \sum D_w$ over $W = U - V$

If P_v is the portion of the original user requirements already covered in the standard commercial package, or

$P_v = V / U$, then

$I = (1 - P_v) * D_u$ and

$$K_v = (D_v * M) / (N * S_v) + (1 - P_v) * D_u \quad (2)$$

Notice that the cost of open source software to a user is the marginal cost of developing additional features, whereas the cost of commercial software licenses is an average cost depending on $(D_v * M) / (N * S_v)$. Implementation is always a marginal cost. Depending on whether licensing or implementation costs are a higher proportion of total costs, commercial software could either be a marginal or average cost proposition to the user. In contrast, open source software is always a marginal cost proposition.

ADOPTION OF OPEN SOURCE VS. COMMERCIAL SOFTWARE

Since rational value-maximizing users would choose open source software over commercial alternatives if $K_v > K_{os}$, we can combine (1) and (2) to obtain the following relationship: Users will choose open source software if

$$(D_v * M) / (N * S_v) + (1 - P_v) * D_u > (1 - P_{os}) * D_u$$

This expression can be rearranged as:

$$(D_v / D_u) * (M / N * S_v) > (P_v - P_{os}) \quad (3)$$

Alternatively, the adoption of open source software would occur when

$$P_{os} > P_v - (D_v / (D_u * N * S)) * M \quad (4)$$

This equation suggests that the adoption of open source versus commercial software will be influenced by the following key factors:

- The relative feature set of open source and commercial software, or P_v - P_{os}. All else equal, the product with the higher feature set would find greater adoption.

- Higher vendor profit margins, however, could lead to the mass adoption of a less feature-rich open source alternative by driving up the licensing costs of commercial software.
- The relative feature sets of what a user actually needs versus what a commercial software package offers, defined by D_v / D_u. If the commercial software product includes many features which are not needed by users, then a less feature-rich open source alternative could see mass adoption.[3]
- The user base of the commercial software product. A higher potential user base (N) or adoption rate (S_v) would help offset more costly feature sets or higher profit margins and raise the threshold for an open source competitor.

As an example, consider two hypothetical software markets. Exhibit 1 shows a hypothetical niche business application. The commercial software package has a relatively low out-of-the-box feature coverage, high implementation costs, and small market size. These factors together create an environment where an open source competitor with a significantly lower feature set could be adopted *en masse*.

In contrast, Exhibit 2 illustrates a hypothetical consumer software package. The commercial product has a higher out-of-the-box feature coverage, very low implementation costs, and a large user base against which development costs and overhead could be amortized. This combination creates a much higher threshold for an open source product to see broad based adoption, despite the vendor's higher profit margins. This example agrees with the conclusions of Casadesus-Masanell and

[3] Note that this relative feature set term has a similar effect as profit margin: a vendor could always offset a higher feature set by reducing the prices of its licenses and accepting a lower profit margin. Indeed, some commercial software vendors have responded to open source alternatives by giving away a free version of their software.

Ghemawat (2006), which showed that a commercial operating system could displace an open source competitor by relying primarily on its larger user rate.

COMPETITIVE VIABILITY OF OPEN SOURCE VS. COMMERCIAL MODELS

Equation (4) also shows that users would be indifferent between open source and commercial alternatives under the following condition:

$$P_{os} = P_v - (D_v / (D_u * N * S)) * M \quad (5)$$

This relationship could be viewed in the context of the scalability of the underlying software product. "Scalability" in this case refers to how easily the same software could meet the needs of additional users. If many more users could use a software package with minimal additional features, then the software package could be called "scalable." Conversely, if each additional user required new features, then the software would not be very scalable. For this analysis, let

$$R = (D_u * N * S) / D_v \quad (6)$$

represent the scalability of the software. Larger values of R would imply greater scalability. The example in Exhibit 1, for example, has an R of 18, while the example in Exhibit 2 has an R of 9,000. Then equation (5) could simply be re-written as:

$$P_v - P_{os} = M / R \quad (7)$$

In particular, the left hand side of equation (7) is the feature advantage a commercial software needs to maintain over an open source alternative, and the right hand side is a combination of the scalability of the software's feature set and the software vendor's profit margin. As M / R increases, either because of higher profit margins at the software vendor or because of lower scalability of the underlying feature set, $P_v - P_{os}$

must also increase to compensate. In other words, the vendor must offer a greater feature set to compensate for a higher profit margin or for lack of scalability of those features.

On the other hand, as M / R decreases, either from greater scalability of features or from a lower profit margin, $P_v - P_{os}$ would decrease as well. At the extremes, $P_v - P_{os}$ could either be 0, meaning that the commercial software package needs to maintain no real feature advantage to be viable to an open source competitor, or 1, which means that the commercial software package is not viable against any open source competitor.

Exhibit 3 illustrates the values of $P_v - P_{os}$ for various combinations of M and R. It shows the required feature advantage a commercial software product must have versus open source alternatives for different combinations of vendor profit margin and feature set scalability ratios. Exhibit 4 illustrates the same relationship in graphical form.

Finally, equation (7) can be re-arranged to derive the maximum sustainable profit margin of a commercial software vendor given a pre-existing feature gap and feature set scalability ratio:

$$M = (P_v - P_{os}) * R \qquad (8)$$

Equation (8) shows that either a larger feature gap or a higher scalability factor would ultimately permit a greater profit margin for a commercial software vendor. This relationship is shown in Exhibit 5 and graphically in Exhibit 6.

CONCLUSIONS

This analysis examined the fundamental competitive dynamics of open source and commercial software based on a model of rational software adoption. It showed that the hurdle for adoption of open source software is a function of the relative feature sets, size of user base, scalability of features, and profit margin of competing commercial products.

Where existing commercial products enjoy a large user base, can address a high portion of user needs out-of-the-box, and the profit margins of the vendor are reasonable, open source software must offer comparable or better features to gain mass adoption. If, on the other hand, user needs are highly heterogeneous, the commercial product's vendor has excessive profit margins, or where the software is fundamentally not scalable, then an open source product with significantly lower feature sets could gain mass adoption.

These conditions suggest that open source and commercial software models may be well adapted to separate niches of the software industry. Open source software may fundamentally be more suitable for software whose features are not scalable, whereas commercial software may be better suited for mass market software with highly homogeneous user requirements and features.

The equilibrium conditions shown here also suggest that the two models serve to check and balance each other. Commercial software sets a threshold of features and a ceiling on costs for new open source products. Open source software, in turn, serves to limit excessive profits and uneconomic feature bloat in commercial products.

Many interesting questions remain about open source software. For example, could empirical studies validate the conclusions derived here? How do users' forward expectations of open source and commercial software development affect their choice of one versus another? Finally, is the equilibrium between open source and commercial software stable over time? These are questions which should be studied to help us better understand the economics of free software.

EXHIBIT 1.

Pv: Vendor Feature Coverage	50%
Dv / Du : Vendor vs. User Requirements	100
M: Vendor profit margin	3
N: Number of potential users	5,000
Sv: Vendor market penetration	35.0%
Pos: Open Source Feature Threshold	32.9%

EXHIBIT 2.

Pv: Vendor Feature Coverage	95%
Dv / Du : Vendor vs. User Requirements	10,000
M: Vendor profit margin	20
N: Number of potential users	100,000,000
Sv: Vendor market penetration	90%
Pos: Open Source Feature Threshold	94.8%

EXHIBIT 3.

Pv – Pos: Required Commercial Software Feature Advantage versus Open Source Competitor

M: Profit Margin \ R: Scalability	10	25	50	100	500	1,000	5,000	10,000
2	20%	8%	4%	2%	0%	0%	0%	0%
5	50%	20%	10%	5%	1%	1%	0%	0%
10	100%	40%	20%	10%	2%	1%	0%	0%
25	100%	100%	50%	25%	5%	3%	1%	0%
50	100%	100%	100%	50%	10%	5%	1%	1%
100	100%	100%	100%	100%	20%	10%	2%	1%

EXHIBIT 4.

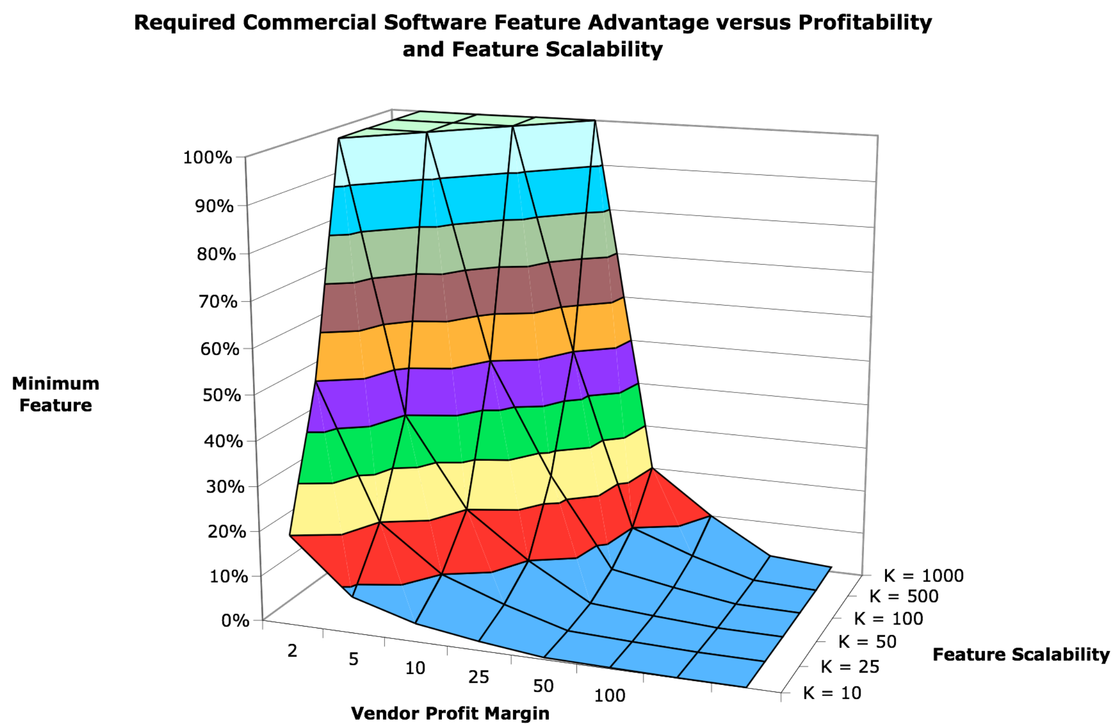


EXHIBIT 5.

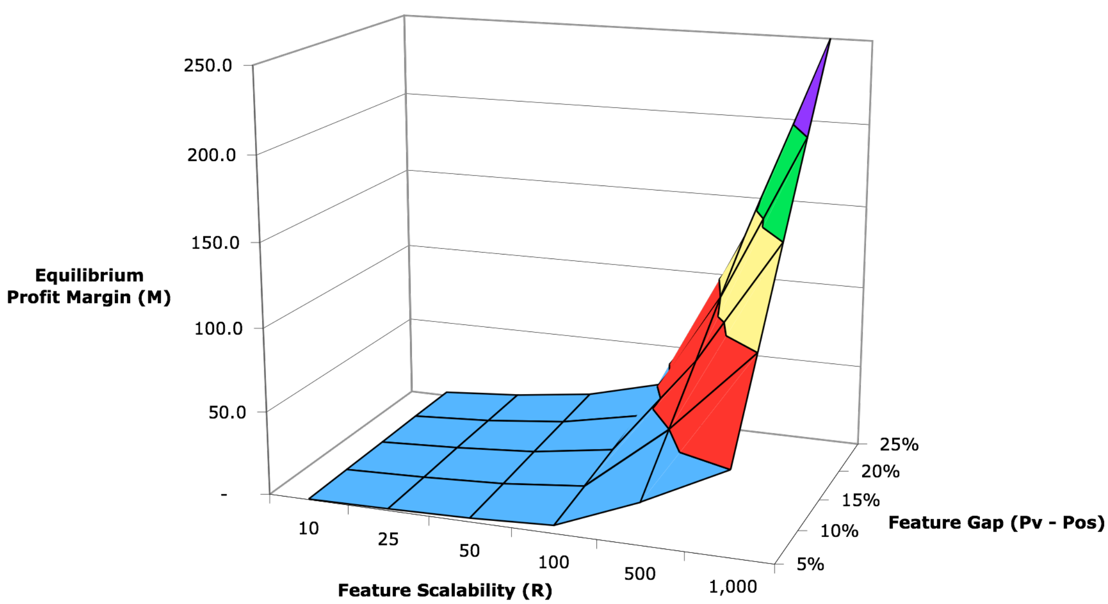
M: Equilibrium Profit Margin of a Commercial Software Vendor for a Given Feature Gap and Scalability Ratio

R: Scalability of Features

Feature Gap: Pv - Pos	**10**	**25**	**50**	**100**	**500**	**1,000**	**5,000**	**10,000**
5%	0.5	1.3	2.5	5.0	25.0	50.0	250.0	500.0
10%	1.0	2.5	5.0	10.0	50.0	100.0	500.0	1,000.0
15%	1.5	3.8	7.5	15.0	75.0	150.0	750.0	1,500.0
20%	2.0	5.0	10.0	20.0	100.0	200.0	1,000.0	2,000.0
25%	2.5	6.3	12.5	25.0	125.0	250.0	1,250.0	2,500.0

EXHIBIT 6.

Equilibrium Vendor Profitability given Feature Set and Scalability of Features



REFERENCES

Axelrod, Robert. *The Complexity of Cooperation: Agent-Based Models of Competition and Collaboration*. Princeton University Press, 1997.

Casadesus-Masanell, Ramon, and Pankaj Ghemawat. "Dynamic Mixed Duopoly: A Model Motivated by Linux vs. Windows," *Management Science* 52, no. 7 (July 2006), pp. 1072 – 1084.

Hertel, Guido, Sven Niedner, Stefanie Herrmann. "Motivation of Software Developers in Open Source Projects: An Internet-based Survey of Contributors to the Linux Kernel," *Research Policy*, 32(7), 2003, pp. 1159 – 1177.

Lakhani, Karim R. and Robert G. Wolf. "Why Hackers Do What They Do: Understanding Motivation and Effort in Free/Open Source Software Projects," *Perspectives on Free and Open Source Software*, MIT Press, 2005.

Koru, A. Gunes and Jeff Tian. "Defect Handling in Medium and Large Open Source Projects," *IEEE Software*, July 2004, pp. 54 – 61.

Lerner, Josh and Jean Tirole. "Some Simple Economics of Open Source," *The Journal of Industrial Economics*, 2 (L), 2002, pp. 197 – 234.

Lerner, Josh and Jean Tirole. "The Economics of Technology Sharing: Open Source and Beyond," *Journal of Economic Perspectives*, 19, 2005, pp. 99 – 120.

T. R. Madanmohan, Rahul De. "Open Source Reuse in Commercial Firms," *IEEE Software*, November 2004, pp. 62 – 69.

Mockus, Audris, Roy T Fielding, and James D. Herbsleb. "Two Case Studies of Open Source Software Development: Apache and Mozilla", *ACM Transactions on Software Engineering and Methodology*, Vol. 11, No. 3, July 2002, pp 309 – 346.

Norris, Jeffrey S. and Poul-Henning Kamp. "Mission-Critical Development with Open Source Software: Lessons Learned," *IEEE Software*, January 2004, pp. 42 – 49.

Ortiz, Sixto Jr. "Embedded Oss Gain the Inside Track," *Computer*, November 2001, pp. 14 – 16.

Paulson, Linda Dailey, "Open Source Databases Move into the Marketplace," *Computer*, July 2004, pp. 13 – 15.

Sacchi, Walt. "Free and Open Source Development Practices in the Game Community," *IEEE Software*, January 2004, pp. 59 – 66.

Spinellis, Diomidis and Clemens Szyperski. "How is Open Source Affecting Software Development?" *IEEE Software*, January 2004, pp. 28 – 33.

Weber, Steven. *The Success of Open Source*. Harvard University Press, 2004.