Business models with free software

Irene Fernández Monsalve

PID_00145049

UOC
Universitat Oberta de Catalunya
www.uoc.edu

Index

Introduction

In the previous modules, we looked at the software market, the traditional types of company in the sector and the possibilities offered by free software in this framework. In this module, we will study the most common business models built around free software, together with some specific cases.

There can be no doubt that free software is emerging as a key element of new business models. After the bursting of the technology bubble (popularly known as the dot-com bubble) at the turn of the decade, free software has driven the creation of new companies in the technology sector, attracting increasing amounts of venture capital. In 2004, a total of $149 million was invested in 20 new companies. In 2006, this amount had risen to $475 million, distributed among 48 business initiatives.

Recommended reading

For more information, see: Larry Augustin (2007). "A New Breed of P&L: the Open Source Business Financial Model". *Open Source Business Conference (OSBC)*.

http://www.osbc.com/live/images/13/presentation_dwn/A_New_Breed_of_P_and_L.pdf

Established and consolidated companies, such as Red Hat or MySQL, have been joined by a new generation of numerous companies whose strategies focus on the use and development of free software. Over the coming years, we will witness the real development of these new businesses and see whether their business model proves sustainable in the long run.

First generation	Second generation	Third generation
Publicly traded: Red Hat, Caldera (now SCO), VA Linux (now VA Software), Turbolinux **Taken over**: SUSE, Cygnus **Other**: LynuxWorks, Linuxcare (now Levanta), Sendmail	**Publicly traded**: Trolltech, Sourcefire, Mandrakesoft (now Mandriva) **Taken over**: Conectiva, Lycoris, JBoss, Sleepycat, Ximian, Gluecode **Other**: MontaVista, MySQL, Zend	ActiveGrid, ActiveState, Alfresco, BitRock, Black Duck, CollabNet, Collax, Compiere, Covalent, DB4O, Digium, Exadel, eZ Systems, Fonality, Funambol, Groundwork, Hyperic, Ingres, Interface21, JasperSoft, Joomla, Laszlo Systems, Medsphere, Mozilla Corp, MuleSource, OpenBravo, OpenLogic, OpenXchange, OTRS, Palamida, Pentaho, rPath, SnapLogic, Sourcelabs, Spikesource, SQLite, WebYog, SugarCRM, Talend, Terracotta, Ubuntu/Canonical, Vyatta, WSO2, XenSource, Zenoss, Zimbra, Zmanda, etc.

Business models with free software: success stories. Table and investment data taken from Marten Mickos (2007). "Open Source: why freedom makes a better business model". *Open Source Business Conference (OSBC)*. http://www.osbc.com/live/images/13/presentation_dwn/Keynote-Open_Source_Why_Freedom.pdf)

In this module, we will study some of the businesses listed in the table above, along with others that are still significant even though they do not attract much attention due to their small size. We will look at the advantages of the free software they exploit, the problems that they have encountered and how they have resolved them. We will also examine various taxonomies to characterise these models and try to identify diverse key factors that determine the operation of the company according to different authors.

Objectives

After completing this module, students should have achieved the following aims:

1. To understand the main classifications drawn up to date for free software models.

2. To know the current business models based on free software.

3. To understand the different mechanisms for revenue generation and differentiation exploited by these models.

4. To be capable of analysing how the different companies use free software to create a competitive advantage.

1. Characterising business models with free software

When we talk about business models based on free software, we are often referring to the new and ingenious ways of earning income that are being implemented, since the traditional model, the selling of a proprietary product, is no longer so clear cut. Companies, in contrast to individuals, need to consider an important factor when they take part in a free software project: how to obtain the economic return that will justify their investment.

In previous modules, we saw how the idea that the income generated by software is directly related to its sale is not an accurate picture of the reality. Most software is developed internally and the sale of software is only the main source of income for a handful of companies. In most cases, it is necessary to offer complementary services to ensure the continuity of income and the survival of the business in harsh times.

Moreover, in the article by Perens that we looked at in the second module ("The Emerging Economic Paradigm of Open Source"), we saw that free software offers much better economic prospects (cost and risk) than the proprietary alternatives for companies that need to develop non-differentiating software.

Recommended website

For more information on Perens:

http://www.uic.edu/htbin/cgiwrap/bin/ojs/index.php/fm/article/view/1470/1385

In all events, this module will show how different companies manage the intellectual property of their products, also generating **mixed models** in an attempt to reconcile the advantages of free models with the generation of direct financial returns based on intellectual property. In this case, the choice of licence will largely determine the range of business models that a company can implement.

2. Classifications according to different authors

In this section, we will study the various attempts to classify business models in literature, pausing to look at the factors that each author has considered crucial for the grouping of the different models. In addition to more theoretical approaches, we will look at those based on observing existing businesses in a more qualitative way and a quantitative methodology for the classification of business models in the context of the **FLOSSmetrics project**. Lastly, we will propose a taxonomy of our own that combines all of the proposals discussed.

2.1. Hecker and Raymond classifications

One of the first authors to write about the business prospects of free software was Frank Hecker in 1998 with "Setting Up Shop: The Business of Open-Source Software". In his article, he takes four OpenSource.org categories and adds others, analysing them on the basis of:

Recommended website

For more information, see:
http://hecker.org/writings/setting-up-shop

- Which companies implement this model?
- What types of licence are appropriate?
- What opportunities for differentiation does the model offer?
- What opportunities does the model offer to set prices based on perceived value rather than on actual costs?

The table below summarises this classification, adding another characterisation parameter, which, though not expressly mentioned by Hecker, is a key feature: how is the company revenue generated?

Model	Source of revenue	Type of licence	Opportunities for differentiation	Price opportunities based on perceived value vs. costs	Cases
Support sellers	Sale of related services (covers all types of services, from custom developments to training, consulting, etc).	GPL	Quality, price, and simplifying and improving the user experience.	Limited. Possible if it has a good reputation.	Cygnus Solutions Red Hat Caldera
Loss leader	Sale of other proprietary products	BSD or Mozilla	Based on the product.	Possible.	Sendmail Netscape
Widget frosting	Sale of hardware		Based on hardware: functionality, performance, flexibility, reliability, cost...	Limited. The hardware pricing system is typically based on costs.	Corel VA Linux

Summary of classification of business models ("Setting up shop: the business of open source" Hecker, 1998)

Model	Source of revenue	Type of licence	Opportunities for differentiation	Price opportunities based on perceived value vs. costs	Cases
Accessorising	Sale of physical products (books, etc).		Product quality (books, etc.) and loyalty from "pro-free software" users.	Limited. Brand reputation can allow prices to be raised slightly.	O'Reilly & Associates
Service enabler	Sale of on-line services provided by the program	GPL or Mozilla	Back-end attributes, creation of unique and useful services.	Possible if a unique and inimitable service is created.	Netscape
Sell it, free it	As a cyclical "loss leader"	BSD or Mozilla	Software functionality (while it remains closed).	Possible until the product becomes an interchangeable asset (at which point, it is released)	–hypothetical–
Brand licensing	Sale of name rights. The version co-exists with the "generic" branded version.		Value added, for example, through additional validation and testing of the non-brand product.		–hypothetical–
Software franchising	Sale of franchise and percentage of franchise revenue		As a support-seller and brand licensing	Possible if it has a good reputation.	–hypothetical–
Hybrids (licences are neither free nor pure proprietary)	Limit code availability: sale of licences under certain conditions				Trolltech Qt
	User-based treatment on – sale to commercial users				Open Group
	Treatment based on use – sale for commercial use, or sale for use on certain platforms				Qt

Summary of classification of business models ("Setting up shop: the business of open source" Hecker, 1998)

In *The Black Cauldron*, Eric S. Raymond also outlines the role of free software in business, focusing, among other aspects, on how free software affects the "**use value**" (**value as an intermediate product**) and "**sale value**" (**value as the end product**) of the software, proposing a taxonomy based on which of the two the company exploits.

Recommended website

For more information, see:
http://catb.org/~esr/writings/magic-cauldron/

For Raymond, only sale value is affected by a free software model, so his classification describes models based on use value and models based on indirect sale value, in which free software makes the sale of another product or service viable:

- Models based on use value
  - Cost sharing (for example, Apache)
  - Risk sharing (for example, Cisco)

- Models based on indirect sale value
  - Loss-leader/market positioner
  - Widget frosting
  - Give away the recipe, open a restaurant
  - Accessorising
  - Free the future, sell the present
  - Free the software, sell the brand

– Free the software, sell the content

As we can see, in the models based on indirect sale value, Raymond includes those of Hecker, plus one new one "Free the software, sell the content". In this model, the value lies in the information provided by the software platform, which is the information sold through subscriptions. The software is released, meaning that it can be carried over to different platforms, thus expanding the potential market of the real product: **the content**.

Though he proposes it only as a hypothetical model, Raymond anticipates the "**social website**" concepts and paradigm shift proposed by O'Reilly in his article "Open Source Paradigm Shift."

Recommended website

For more information on "Open Source Paradigm Shift" see:
http://www.oreillynet.com/pub/a/oreilly/tim/articles/paradigmshift_0504.html

However, he does not recognise the role of the Internet as a platform or the subsequent "software as a service", considering that the value of releasing the software will lie in carrying it over to other platforms, thus contributing to its diffusion and market expansion.

2.2. European Working Group on Libre Software

The business models presented by Hecker and Raymond are based on observation of companies that used free software as part of their business models, though they perhaps lack a degree of systematisation and abstraction in their taxonomy. In its document "Free Software/Open Source: Information Society Opportunities for Europe?", the **European Working Group on Libre Software** (http://eu.conecta.it/paper/) makes an analysis based on how free software projects are funded rather than on the basis of their business models and regardless of whether the project is linked to a specific company:

- Public funding.
- Non-profit private financing.
- Financing for those who need improvements.
- Financing with related benefits (O'Reilly and Perl).
- Financing as internal investment.
- Other (bonuses, development cooperatives, use of markets to establish contact between clients and developers).

Its "Financing as internal investment" section, however, contains a classification of business models, which include, among others, the possibility of generating **revenue through services**, as a result of the competitive advantage afforded by being the main developers of a given software project.

Model	Differentiation	Revenue	Licences	Examples
Better knowledge here	Better understanding of the product: must be the developer of the product or a collaborator.	Related services: custom developments, adaptations, installation, integration.	Free	LinuxCare (in its early days) Alcove
Better knowledge here with constraints	Better understanding of the product: must be the developer of the product. Part is kept proprietary.	Related services and sale of proprietary part.	Free and proprietary	Caldera Ximian
Source of a free product	Producer, almost entirely free product.	Related services: custom developments, adaptations, installation, integration.	Free	Ximian Zope Corporation
Source of a free product with constraints	Proprietary product in principle. Subsequent release as a strategy to expand adoption and other advantages of free software.	Sale of commercial version.	Free and proprietary	Artofcode LLC Ada Core Technologies
Special licences	Best knowledge here Offer of proprietary version for customers who do not want GPL.	Sale of commercial version, and related services.	GPL and proprietary	Sleepycat
Sale of brand	Based on image and brand, allowing the product to be sold at a higher price.	Sale of distributions, and related services (including certification and training)	Free	Red Hat

Business models based on free software. ("Introduction to Free Software" teaching manual)

2.3. Empirical studies

In "Business models in FLOSS-based companies", Carlo Daffara describes an empirical study of business models based on the use of free software, undertaken in the context of the **FLOSSmetrics project**. The study also examines how these models handle the marketing of their products and what licences they use.

Recommended website

For more information, see:
http://opensource.mit.edu/papers/OSSEMP07-daffara.pdf

The study started out with 120 companies, of which it eliminated those not considered to be based on FLOSS (free, libre and open source software), and those that only allowed access to the code to non-commercial users or which did not allow redistribution. It also eliminated companies that, despite making important contributions to free software projects, do not base their core business model on it (such as IBM, HP and SUN).

It selected a set of characterising features, such as licensing, products and services offered (installation, integration, training, consulting, legal and technical certification), types of contract (subscription, licence, or per-incident) and self-referential literature offered on their websites and information on their relationship with the community. Lastly, the data were collected and all non-significant variables were eliminated to obtain the following characterising variables:

- Main revenue generator
  - Selection.

  - ITSC (installation/training/support/consulting). The different types of service are grouped together, since the study found that the companies offering one also tended to offer the others.
  - Subscriptions.
  - Licences.

- Licensing model

Applying cluster analysis to the companies characterised by these variables, the study obtained six basic business models, and a seventh group that was analysed separately:

1) **Twin licensing:**dual model of GPL and proprietary licence in order to sell to those who want to develop closed-source code based on the free product.
2) **Separate OSS and commercial products:** sale of commercial products based on a free one.
3) "**Badgeware**"**:** brand protection; released products must keep original logo/authorship visible.
4) **Product specialists:** creation of a free product and sale of services relating to it.
5) **Platform providers:** selection, integration and support services, providing tried and tested platforms.
6) **Selection/consulting companies:** generic services and analysts do not generally contribute to the community, since the results of the analysis and consulting are kept private.
7) **Ancillary markets:** by way of example, SourceForge/OSTG generates most of its revenue from sales from its affiliate site, ThinkGeek. Although this model is not one of those characterised by the study (the limited number of cases in this category did not allow for extrapolation), it should not be underestimated as it is an important financing model.

The following table shows the results of the study.

		Main Licensing model					Main revenue generation			
	Company	twin licensing	OSS and commercial versions	Badgeware	Pure OSS	multiple packages covered	selection	ITSC	Subscription	licensing
twin lic.	Funambol	•						•		•
	Lustre	•						•		
	MuleSource	•							•	•
	Mysql	•							•	•
	OpenClovis	•							•	
	Pentaho	•						•		•
	sleepycatdb	•								•
Split OSS/ commercial releases	Adaptive Planning		•							•
	Alterpoint		•					•		•
	Altinity		•					•		•
	Codeweaver (WINE)		•							•
	Coupa		•							•
	Digium (Asterisk)		•						•	
	Enormalism		•							•
	EnterpriseDB		•							•
	GreenPlum		•							•
	GroundWork		•						•	
	Hyperic		•						•	
	JasperSoft		•							•
	Knowlede Tree		•	•						
	OpenCountry		•							•
	Open-Xchange		•							
	NoMachine NX		•							•
	rPath		•						•	
	Scalix		•							•
	Sendmail		•							•
	Smoothwall		•						•	
	Sourcefire (SNORT)		•						•	
	Splunk		•						•	
	SSLExplorer		•							•
	SugarCRM		•	•						•
	TenderSystem		•	•						•
	VirtualBox		•							•
	Vyalta		•					•	•	
	XenSource(Xen)		•					•		
	Zen(PHP)		•							•
	Zimbra		•	•						•
Badgeware	1bizcom			•				•		
	CATS applicant tracking			•					•	
	EmuSoftware/Netdirector			•				•	•	
	Jbilling			•				•		
	OpenBravo			•				•		
	OpemEMM			•				•		
	Open Terracotta			•					•	
	SocialText			•						•
product specialists	Alfresco				•			•	•	
	Babel				•			•		
	CentraView				•			•		
	CleverSafe				•			•		
	Compiere				•			•	•	
	Exadel				•			•		
	Jitterbit				•			•	•	
	Mergere				•			•		
	Mindquarry				•			•		
	Mirth				•			•		
	OfBIZ				•			•		
	Qlusters (OpenQRM)				•			•		
	Symbiot/OpenSIMS				•			•		
	Talend				•			•		
	UltimateEMR				•				•	
	VISTA				•			•		
	vTiger				•			•		
	Zenos s				•				•	
Platf. Prov.	Jbos s				•	•		•	•	
	RedHat Linux				•	•			•	
	Sourcelabs				•	•	•		•	
	SpikeSource				•	•	•		•	
	SUS Linux				•	•			•	
	WSO2				•	•		•		
selection/consu lting	ayamon					•	•	•		
	Enomaly					•	•			
	navica					•	•			
	openlogic					•	•			
	Optaros				•	•	•			
	x-tend					•	•	•		
Other	CiviCRM				•					
	Eclipse				•					
	Mozilla				•					
	OSAF Chandler				•					
	Sourceforge									

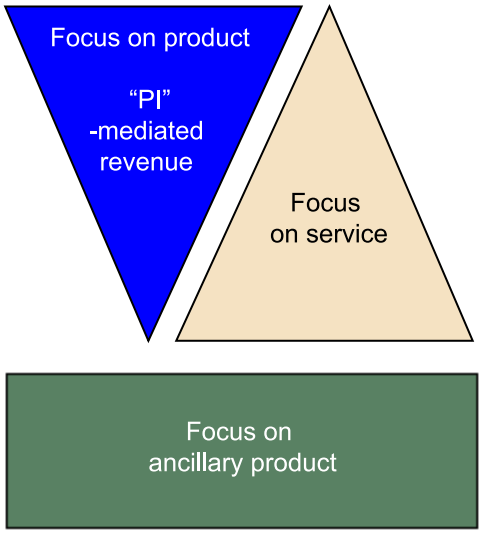
Business models in **FLOSS** companies (Carlo Daffara, "Business models in FLOSS-based companies http://opensource.mit.edu/papers/OSSEMP07-daffara.pdf)

2.4. Proposed classification

The last classification analysed is interesting because it provides empirical data on real companies that currently focus their business model on free software. However, like Hecker, Daffara proposes a characterisation in isolation, rather than a taxonomy. We now propose a schema of our own to sort and incorporate the ideas we have analysed thus far, classifying the models by the degree to which their revenue is derived from the intellectual property rights over the software and by the extent to which they focus on the provision of products or services:

- Specialist/Vertical
  (Development companies whose main product is
  a free software programme)
– Mixed OSS/proprietary: dual licenses
- Mixed OSS/proprietary: free kernel, proprietary accessories
- Purely OSS: "Distributed sale" of free product
- Purely OSS: Services on product
- Software as a service (SaaS)

- Providers of services associated with the software
- Platform distribution companies
- Large integrators
- SMEs and niche micro-companies
- Ancillary markets
- Hardware
- Other



Our classification, like that of other authors, is based on source of revenue. Nonetheless, besides considering how the different companies recover their investment in free software development, it is also important to analyse how they exploit the advantages that a free development model can offer.

Business models are also characterised by their source of revenue, by the market they are aimed at, how they develop and market their products, and by how they relate to the competition. Hence, there is a cross-cutting issue affecting any business model that becomes particularly relevant with the use of free software: the concept of **coopetition**.

Coopetition

Among the other features differentiating free and proprietary software is the fact that the use of free software can enhance the quality of the services offered, thus helping to remove barriers to entry and sketching out a scenario of increased competition and effort for differentiation and specialisation, besides a distinct, open, cooperative competition in which companies will need

to cooperate as well as compete if they wish to prosper. This business concept, which in some ways is replacing that of "winner takes all" in the context of a new network economy, is called coopetition.

Coopetition: cooperation between competing companies to seek win-win scenarios, either to enhance the value of the product or to expand the market.

Recommended website

For more information, see:

Henry Chesbrough; Wim Vanhaverbeke; Joel West. "Open Innovation: researching a new paradigm"

http://www.openinnovation.net/Book/NewParadigm/Chapters/index.html

In this context, companies need to carefully examine their **economic ecosystem** – clients, providers, competitors and complementers – implementing strategies for the creation of new alliances and rethinking their traditional associations.

This concept is not unique to free software and has extended to other areas. Companies in the same industry can collaborate with one another to expand their markets, competing later when it comes to segmenting them.

Example

Intelwill invest considerable sums in expanding the microprocessor market, even though part of this investment will directly benefit its competitor, AMD. In this case, given Intel's dominant position, the percentage of its investment that will benefit others will be quite low.

Although coopetition is not exclusive to free software, it is highly significant in open-source development scenarios. It is inevitable that the competition will benefit from our investment, so it is necessary to find ways to turn this apparent disadvantage into a business advantage. Moreover, incorporating the users (clients) into the development process, involving them and encouraging their participation as allies, is also a feature of the free software development model.

To a large extent, the use of free software will also limit the possibility of becoming a monopoly and provide an anti-captivity guarantee. Again, a key question for any company becomes especially relevant in free software scenarios: how can we create value for a client while at the same time extracting some of this value for the company?

3. Business models with free software

In this section, we will study each business model with specific examples. Note that these are not rigid models, but rather a diffuse continuous sequence. Many of the companies that we will mention combine several of the models, although we put them into categories for their systematic study.

3.1. Specialist/vertical (a free application as the main product)

In this section, we include companies that produce free software as the promoters and/or leaders of specific projects. Their involvement with free software is thus very significant and one of the key aspects of their business strategy will be the management of the community and seizing of the opportunities for innovation, diffusion and volunteer work that it offers. In essence, these models have a free product for the community and a product or related service as their commercial offer, so the key to their success is often to strike a balance between the two. According to Marten Mickos, CEO of MySQL AB:

> FOSS companies will not work unless they serve equally those who want to spend time in order to save money, and those who want to spend money in order to save time".

These companies are the most common in Daffara's study, including the first four categories (twin licensing, OSS/proprietary versions, badgeware and product specialists). They equate to the product companies we saw in module 3 of this subject, so their main problem will be how to recover the initial investment in development.

As we saw in the above classifications, a common strategy is to obtain revenue through proprietary licences, which are combined with free licences in different ways.

There are also **models that cyclically combine proprietary licences**, like Hecker's "loss leaders" and "sell it, free it". **The loss-leader concept is not unique to software**, being a widespread strategy in every sector of activity: a product is offered free of charge – or at such a low price that it entails losses for the supplier – as a way of attracting the attention of a large number of potential customers to whom the company intends to sell other items. Hence, both dual licence models and free products with proprietary extensions use a loss-leader strategy to some extent.

Besides promoting the sale of the related product, there are several benefits to adopting an open-source strategy of this nature in the software industry, such as helping to establish the technology as a *de facto* standard, attracting

improvements and complements to make the product more appealing, generating sympathy in an audience that includes potential customers of the related product, and reducing the maintenance costs of the project.

We will now look in detail at the twin- or dual-licensing model and the model combining a core free product with proprietary accessories. We omit other models in which the main product is not free because they are really business models based on proprietary software: the code is only released as a complementary business strategy to enhance the position of the core proprietary product.

Daffara also lists several companies that carry out development projects with entirely free licences and earn their income from **ITCS**(installation/training/support/consulting). This group is perhaps one of those that can encompass the most different models, since its revenue source is a rather vague category. Hence, it is important to look closely at the markets they serve and their differentiation with equivalent products, in addition to best knowledge.

3.1.1. Mixed models: dual licensing

This model is based on the distribution of a product under two different licences: a traditional proprietary licence and a restrictive free licence (GPL type). Thus, if somebody wishes to derive a work from it and redistribute the new work without the code, they can, but they must pay for a licence. Otherwise, all derivative works must be redistributed with the code.

Michael Olsen, manager of Sleepycat Software Inc., producers of BerkeleyDB, describes its dual-licensing model thus:

> "The Sleepycat open source license allows the use Berkeley DB [...] without cost, under the condition that if the software is used in an application that is later redistributed, the complete code of the application must be available, and must be able to be redistributed again freely under reasonable conditions. If you do not want to offer the source code of an application derived, you can buy a Sleepycat Software license."
>
> S. Comino; F. M. Manetti. "Dual licensing in open source markets". Available at: http://opensource.mit.edu/papers/dual_lic.pdf

This strategy is appropriate when a substantial proportion of the demand is generated by commercial users who need to embed the software in their own products. These customers use the product purchased as input for the production of new software, either as an end product or as part of a more complex technology produced and sold by the commercial customer. Whether because they need to be able to sell their derived products under a traditional propri-

etary system or because the software they generate is a fundamental part of their differentiation, this customer will need to close the code it generates and must therefore pay to do so.

These models divide their users into two groups: the community – all users who are content with free licences, and use the product under these terms – and corporate clients sensitive to the reciprocal terms of free licences.

However, maintaining a community of people collaborating on the product can be problematic. On the one hand, if direct income is obtained through the product, this could affect the motivation of the volunteers who contribute without receiving anything in return. While on the other hand, the companies that implement it must formally obtain copyright assignment from the volunteers in order to avoid future problems from disgruntled employees claiming their share of revenue from licences for the product that they helped develop.

In practice, companies that base their model on dual licensing do not benefit greatly from the possibilities of external contributions to the development, obtaining only small-scale debugging and the odd patch from the community. The main development team is typically almost 100% dominated by company employees.

Another problem that can arise with these models is that their customers can build their own proprietary extensions without modifying the original code, so they can use the free licence version and have their add-ons as a separate and independent application.

These companies often combine the revenue generated from dual licensing with other activities such as the provision of services, which we will see later. Examples of this model include Funambol, MySQL, Sleepycat DB, and Troll-Tech/NOKIA.

The Funambol case

Company name	Funambol, Inc.
Head office	Redwood City (United States)
Website	www.funambol.com
Creation date	2001
No. of people employed in 2007	40
Turnover in 2007 (million)	$4.8

Corporate data on Funambol, Inc. Table prepared with statistics from Hoovers (http://www.hoovers.com)

Funambolis a US corporation engaged, as its motto states, in "mobile 2.0 messaging powered by open source". The company develops a mobile application server (providing push e-mail, address book and calendar, data synchronisation, and an applications server

Recommended website

For more information, see:
http://www.funambol.com/blog/capo/2006/07/my-honest-dual-licensing.html

for mobile devices and PCs), together with a development platform for mobile applications, both developed under the name "Funambol".

It sells its code base under two licences: the AGPLv3 for its "Community Edition" and a commercial proprietary licence for its "Carrier Edition". It also combines this strategy with providing the additional functionality required for large-scale implementations of the closed version as well as services based on the "Carrier Edition.

In this case, Funambol chose the "Affero" GPL licence, which affords it extra protection against the commercial use of its applications in the form of software as a service (SaaS). As discussed in module three of this subject, the GPL allows the code to be modified without redistribution, provided that the application itself is not redistributed, as occurs with the provision of software as a service. The "Affero" GPL solves the problem of this void by requiring redistribution of the source code when the software functionality is offered under the "SaaS" model too.

The nature of the software makes it an ideal candidate for the dual-licensing model as it appeals to other companies seeking to develop closed applications on its platform, such as mobile telephony operators, device manufacturers and other software companies. Because of its use of the AGPL, companies that use Funambol as the basis of their "SaaS" offers must also pay if they do not wish to redistribute the code. Its customers include Vodafone, Earthlink and Computer Associates.

Funambol tries to fully exploit the needs of large corporate customers with the incorporation of additional functionality in its commercial version and services. To avoid the problems of the "free kernel + proprietary accessories" model that we shall see later, it makes sure that the closed functionality is only interesting in scenarios of large corporate implementations, so its free user community will not feel the need to develop this functionality by itself.

In "My Honest Dual Licensing", Fabrizio Capobianco, manager of Funambol, argues that the dual-licensing model is the most "honest" model in upholding the principles of free software development, making it more compatible with business' need to generate profits.

However, as we saw earlier, defining a viable source of income does not guarantee the success of any company and the use of free software will allow us to implement qualitatively different strategies to those of a model based on proprietary software. Funambol is a case in point here, since the company had to refine its marketing practices and their target populations before hitting on a viable business model.

In its early days, Funambol tried to set up a classic software-vendor model around its free software product. The company developed Sync4j, which enabled developers to build applications for mobile devices under the "sometimes-connected" paradigm (the application can work offline, synchronising data when the connection is restored). It identified large companies and wireless operators as prospective clients which, due to large staff numbers and the increasing opportunities for mobility, would need to synchronise data between mobile devices and their corporate servers.

In order to reach these customers with its product, Funambol decided to pursue a proactive sales strategy with particular emphasis on marketing and its sales force, which tried to access the potential customers directly through telephone campaigns.

It met with very limited success. Funambol failed to meet its sales expectations because it found that large corporations were reluctant to deal with small new companies. In addition, the sales cycles were very drawn out and it soon became apparent that a much bigger sales and marketing team was required to maintain this strategy than Funambol could afford.

Funambol quickly realised that its problems were down to this active sales strategy, the traditional method in the world of proprietary software but a barrier to entry that only a handful manage to overcome: to access a group of potential customers consisting of large corporations, it is often necessary to have a large sales and marketing capacity, as well as a sufficient size and reputation to deliver the required confidence.

The use of free software allowed the company to reverse this strategy, focusing on a reactive type of marketing in response to initiative from the customer. In this new scenario, potential customers would seek out Funambol, leaving the company with the role of being attentive in order to identify them after contact.

The effectiveness of this strategy depended on a single factor: the number of downloads of its product. Once it had obtained enough downloads, it was able to identify the following typical sales cycle (much shorter than the one observed with the previous strategy):

1) The potential user accesses the Sync4j website for information on the product and technical documentation.

2) The user downloads the product.

3) He/she later subscribes to the mailing list for more information.

4) Following extensive use of the product (usually in R&D projects), the customer contacts Funambol to ask about pricing and licence conditions. Internally, they are classified as prospective customers.

5) Finally, they ask for a quotation and formal offer and can become a Funambol client.

(Fabrizio Capobianco; Alberto Onetti. "Open Source and Business Model Innovation. The Funambol case". Available at: http://oss2005.case.unibz.it/Papers/4.pdf)

The key factor for continuing to fuel this cycle is, as we explained earlier, maintaining a high number of product downloads. The cycle is continuous, generating more downloads by itself. Thus, after the initial effort, this mechanism gathers enough inertia to work virtually alone.

To do so, Funambol concentrated on **creating a community around its product**, focusing its marketing efforts on the users of its free version, both experts and those with fewer technical skills. Although this strategy is not directly oriented towards its revenue-generating customers, it proved much cheaper and more efficient.

The company focused on raising the profile of the product among developers, participating in development forums, mailing lists, specialist publications, conferences, creating partnerships with non-profit organisations that promote free software and establishing synergies with other well-established open-source products. For more inexperienced users, it had to ensure that the product was easy to install and that sufficient documentation was available on the website. When it began to focus on these last two factors, the company observed a substantial increase in the number of product downloads, thus setting in motion its sales-generating cycle.

3.1.2. Mixed models: free product kernel and proprietary accessories

In this model (Daffara's "Split OSS/commercial releases"), a program has two different versions: a free basic version and a proprietary commercial version based on the former but with additional functionality implemented through plug-ins or accessories. The free version must use an MPL or BSD type licence allowing the combination in order to create a closed product.

The main problem with this model lies in keeping the free product interesting enough without taking value away from the revenue-generating proprietary product. We also run the risk that the community formed around the product may decide to develop the functionality of the proprietary version on its own, making it difficult to generate revenue from sales.

In this model, we can distinguish between two classes of users: those who are willing to pay for a product with some additional features (medium and large companies), and those who are very sensitive to price, such as small businesses, micro-enterprises and private users. By combining free and proprietary versions, we obtain a more widespread adoption of the proposed solution with-

out missing out on revenue capture from proprietary versions. As we saw in previous modules, in a "winner takes it all" scenario, common in software, strategies based on widespread adoption are very important.

Hence, it is based on the same user-segmentation principles as the dual-licensing model but is more at risk of losing the sympathy of the community, since it does not have access to the entire source code.

An example of this model is **Sendmail Inc.**, which sells an array of proprietary products around the sendmail open server. Other examples include Hyperic (IT Operations/Monitoring), SourceFire (SNORT commercial version), Zimbra/Yahoo (messaging, groupware) and XenSource/Citrix (virtualisation).

The Sendmail case

Company name	Sendmail, Inc.
Head office	Emeryville, CA. (United States)
Website	www.sendmail.com
Creation date	1997
No. of people employed in 2007	125
Turnover in 2007 (million)	$23

Corporate data on Sendmail, Inc. Prepared with statistics from Hoovers (http://www.hoovers.com)

When studying business models based on free software, we often think of corporations that decide to open up their code as a competitive advantage to expand their market share. Sendmail is an interesting case as this process occurs in reverse: with free, non-profit roots, the creation of a commercial initiative around the project is aimed not only at generating revenue from the development, but also to maintain the project's dominant position in its sector and to expand its user base.

Sendmail is a mail transfer agent (MTA) and one of the best known examples of projects born out of free software communities. In 1998, it was estimated that 80% of all e-mail traffic was sent through Sendmail. It is still the most popular MTA on the Internet, although it has lost some users to Microsoft Exchange Server, Exim and Postfix. Equally important is the long lifespan of the product, whose origins date back to developments started in the 1970s.

Eric Allman developed the first version of Sendmail at Berkeley University in the early 1980s on the basis of previous work on the Delivermail program and founded Sendmail, Inc. in 1997. The company strategy focused on selling additional Sendmail functionality in a proprietary format (e.g. user-friendly interfaces) in addition to providing complementary services. At the same time, the company made an effort to openly maintain the continuity of Sendmail's development by providing hosting services and human resources for its development.

When he set up the company, Allman expected not only to develop a business, but also to protect Sendmail's dominant position, which was being threatened by the emergence of proprietary formats that jeopardised the SMTP open standard. The company concentrated its efforts on the corporate environment, offering not only integration and support services, but also a product that was more responsive to its needs. The extensions created by the company provide graphical interfaces and ease of management, and are marketed in proprietary formats.

"Sendmail, Inc. develops commercial products and services for ISPs and enterprises for whom email is mission critical, while continuing to drive innovation and standards through Open Source software development."

Sendmail, Inc

We can consider the creation of Sendmail, Inc. to have been the necessary step to cross the "chasm" and guarantee the product's adoption by the pragmatic and conservative majorities. Nonetheless, for Allman, it was important to maintain the original functionality of free Sendmail, so Sendmail Consortium was set up as a non-profit entity to develop the free version. In this way, it can capitalise on the advantages of an open development model, such as contributions, cost-cutting, product innovation and evolution.

Allman thus took advantage of "the chasm" to sell proprietary extensions to his product without the danger of forking his project. Following Moore's model, the community around the free Sendmail project consists of innovators and technology enthusiasts interested in the raw functionality and new proposals. Business customers, however, are pragmatists and conservatives with very different needs and aims. The proprietary extensions, which focus on the functionality of the product packaging and finish (ease of use, graphic interfaces, stability, etc)., are not only uninteresting to innovators, they may even seem unnecessary. The presence of this chasm between the interests of the community and commercial customers allows for the co-existence of the core free version and the widespread proprietary version without the risk of forks, since the community has no interest in the extensions on the other side of the chasm.

3.1.3. Free models: "distributed sale" of the product

It is commonly assumed that licensing a product in free format leads to loss of opportunity for earning direct revenue from the intellectual property rights over it, creating the need to exploit complementary products or services.

However, choosing a free licence for a project does not necessarily mean forgoing the possibility of obtaining revenue directly from this product. The widespread idea that nobody will pay for something they can obtain for free does not paint a true picture of reality. Many people are willing to pay a small sum for a work that they value if they think that this money will go to the original authors. If a project is successful enough, it may receive small contributions from a lot of people, perhaps even managing to fund its creation in the same way that a street artist does not charge admission but can raise enough to make his or her investment in time and effort worthwhile. This is the idea behind the*The Street Performer Protocol and Digital Copyrights*, by John Kelsey and Bruce Schneier, which proposes a distributed funding mechanism for digital works in which the author does not complete his/her work until sufficient funding has been collected.

Recommended website

http://firstmonday.org/htbin/cgiwrap/bin/ojs/index.php/fm/article/view/673/583

Different mechanisms have been described and implemented for structuring this direct, distributed funding in the context of software development, from grants and bounties to the creation of on-line markets that bring together developers and prospective clients, based on a bonus scheme similar to that described by Chris Rasch in*The Wall Street Performer Protocol*.

Recommended website

http://www.firstmonday.org/issues/issue6_6/rasch/index.html

Donations are the most straightforward mechanism for this type of financing, but too unstable for creators, who need the security of an income before they invest their time. In bonus and bounty systems, the people interested in a specific functionality offer a reward for it to be implemented. When the total reward – which various people can contribute to – reaches a sufficient sum for a developer, he or she can offer to do it and is paid once it is finished. Some of these systems rely on the trust between the development team and the users, and have no payment guarantees, while others propose the establishment of some form of neutral intermediary.

The Cherokee server

This server decided to implement a bounty system with the primary aim of attracting new developers to the project. Besides rewarding effort, providing a financial incentive would attract more people to the development community and encourage the growth of the project.

The key to success in these scenarios may lie more with the payment facilities offered than with the willingness of users to pay:

> "Most people are happy to pay a tiny extra bit on top of some larger amount, if they have their wallet out already and think it's for good reason. When people fail to make small, voluntary donations to a cause they like, it's more often due to the inconvenience (writing a check, putting it in the mail, etc), than the money.
>
> (Karl Fogel. "The Promise of a Post-Copyright World". Available at: http://www.questioncopyright.org/promise)

Virtual markets

Several attempts have been made to create "virtual software markets" based on this type of funding. Some of those currently operating include BountyCounty (http://bountycounty.org/) MicroPledge (http://micropledge.com/) and BountySource (https://www.bountysource.com/).

Although many projects implement these ideas to obtain additional funding, it is difficult to identify corporate scenarios where the bulk of the revenue is obtained through these mechanisms.

Firstly, in the context of software, this type of funding can be more difficult to obtain because of the absence of a strong identification with and sympathy for the authors, which does exist with other creative works.

Secondly, this model is likely to be more successful if it is a non-profit free software project composed entirely of volunteers, which will arouse the sympathies of its users more easily. A company wishing to use it successfully will no doubt have to obtain prior acknowledgement through transparency and trust, proving that profit-making is not the be all and end all and that the project will have an impact on the common good (we will look later at business models based on these principles).

These systems have a more direct economic model, eliminate intermediaries and ensure greater proximity between users and developers. In one sense, they could be considered the natural way to fund a free software project: just as volunteers contribute to varying degrees and in different aspects of the software development cycle, so too can users form part of the project by making a financial contribution in line with their possibilities and interests.

3.1.4. Free product plus associated services

Companies in this category implement a strategy of the type "best knowledge here" and "best code here", developing a free product and offering services for it as a means of generating revenue.

This sections encompasses both the product specialists and badgeware of Daffara's study, since they both represent the same business model. Moreover, although badgeware licences include an additional assignment constraint, they maintain the essential characteristics of openness and freedom of knowledge and can generate the same benefits through their development communities as those that use licences without this constraint. The companies constituting examples of badgeware probably also seek to launch some sort of brand strategy, so they place special importance on assignment when redistributing the products they generate.

This model has a number of problems, such as few barriers to entry to the business – any company can gain knowledge of the product and offer services – and problems obtaining support contracts as client companies may prefer to continue with their regular service or consulting companies or to hire providers that offer support for their entire new technology infrastructure and not just for a specific product.

Another common problem faced by these models for generating revenue from services is that of innovators and enthusiasts: when a new product comes on to the market, its early users are often people with technical skills that will not contract support services for it, preferring to acquire the necessary knowledge for themselves. This model then will need to offer an extended product and transmit reliability in order to reach a potential market that will pay for services relating to the product.

The success of this type of business model is questioned by some authors (like Perens). Nonetheless, there are many companies based on this model that have attracted large sums of venture capital. For a more sustainable business model, however, they will need to address the problems mentioned above.

The models of vertical service provider specialists include Alfresco (content management), Compiere (ERP, CRM), vTiger and Openbravo.

The Openbravo case

Company name	Openbravo, S. L.
Head office	Pamplona (Spain)
Website	www.openbravo.com

Corporate data on Openbravo. (Obtained from http://www.camerdata.es)

Creation date	2001
No. of people employed in 2007	26 to 50
Turnover in 2007	Up to €300,000

Corporate data on Openbravo. (Obtained from http://www.camerdata.es)

Openbravo is an interesting example of this type of model. The company, founded in 2001, develops two free applications for SMEs – OpenbravoERP (enterprise resource planning) and OpenbravoPOS (point of sale) – which seek to meet the needs of management and planning and of point of sale terminals for small and medium enterprises, respectively. The code was published in 2006 and is currently among the most active projects in the SourceForge ranking.

The company has attracted substantial sums of venture capital, with investors such as Amadeus, Gimv, Adara, and SODENA (Sociedad de Desarrollo de Navarra), which has invested €5 million in the company.

Its business strategy is geared towards becoming a leading product in the industry and making OpenbravoERP the benchmark for management software among SMEs. To achieve this, the company is exploiting the possibilities of free software to the maximum through careful community management and application of the coopetition concept.

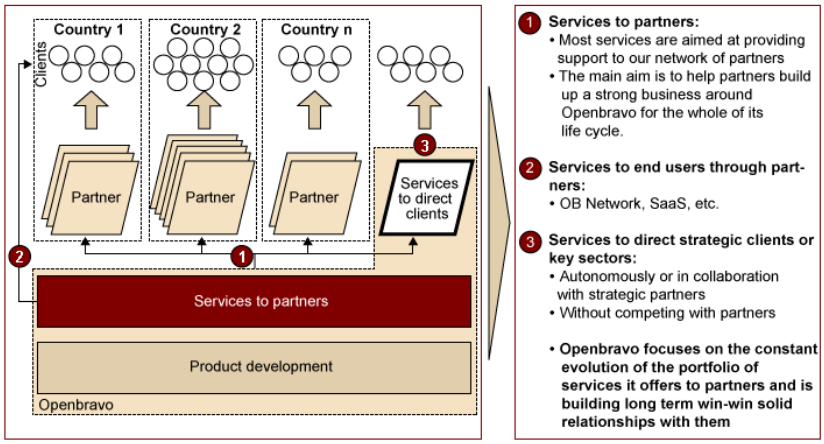
As we saw with Funambol, Openbravo observed that, alone, it did not have the capacity to disseminate and distribute its product among potential users. Although OpenbravoERP and OpenbravoPOS are aimed at SMEs rather than large corporations, in order to achieve its strategic aims of becoming the sector leader, the product had to reach countless small and medium businesses worldwide.

In addition to the size requirements for conducting a marketing campaign of this scale, the company was also aware of the potential difficulties of competing to provide services directly to end users, who might prefer local businesses or ones offering integral solutions and not just companies dealing with a single product.

To overcome these barriers, Openbravo positioned these companies as collaborators rather than competitors. Thus, it admits that, simply because it developed the product, this does not necessarily mean that it is the best company for providing related services to end users. Its mission was to create a good product that could expand markets, generating new revenue opportunities for IT services companies, which could complete its offer with OpenbravoERP and OpenbravoPOS.

Thus Openbravo defines the provision of services to other IT services companies – intermediaries between it and the end users – as its revenue generator. These companies form a network of partners that carry out the tasks of implementation of OpenbravoERP and OpenbravoPOS in SMEs.

Openbravo offers its partners various services (support, training), by implementing a pyramidal system of consulting similar to that described in module 3 of this subject, as well as conveying reliability and trustworthiness. As they are supported by the product developers, they can exploit the strategy of "best knowledge here" and "best code here" on their markets.



Openbravo: Business opportunities and paths for growth (obtained from the Openbravo presentation at WhyFLOSS Madrid 2008 "Openbravo: keys to success in free software application development". http://www.whyfloss.com/es/conference/madrid08/getpdf/49).

Openbravo thus operates a strategy of coopetition, giving service companies the opportunity to exploit Openbravo in the context of their natural markets while benefitting from the increased diffusion of its product, and obtains revenue directly from its partners. Thus far, it has been considerably successful with this strategy and currently has eighty-five partners around the world.

3.1.5. Software as a service

Companies that develop a product can also exploit it through the paradigm of software as a service. Instead of offering installation and support services, the company is responsible for all hardware and software infrastructure, offering functionality directly through the Internet. The recurring revenue generated takes the form of service subscriptions.

Collabnet: software as a service

A good example of this type of model can be seen in **CollabNet**, which provides services for collaborative software development (version control, issue tracking, communication, etc.), generated, among others, through the Subversion version control platform. In this case, in addition to keeping the source code open, the company spends a lot of effort on maintenance of the community, so that its work on the project is merely a contribution – albeit a large one – within a free community. Other examples of companies that market their products according to the "software as a service" model include SugarCRM, SocialText and JasperSoft.

With the "software as a service" format, these companies will not come across any more difficulties generating revenue than their proprietary equivalents, since the sales in this case are not derived from the copyright on the product. The fact that a client can download, install, configure, host and maintain the application will be more a tool for marketing and distribution than a loss of income. As noted earlier, corporate clients are willing to pay for having their problems solved.

Nonetheless, releasing all of the code creates problems with differentiation and opportunities for the entry of competitors. Any company with a sufficient technical capacity and infrastructure could offer a similar service if the code were available. In the light of this problem, the company that developed the product could base its differentiation on "best knowledge here" and "best code here" to gain the sympathy of the community. In addition, if its competitors

also chose to contribute to the development, it could set up coopetition mechanisms, collaborating to expand the market and segmenting it later according to specialisation.

Like the mixed OSS/proprietary strategies we saw earlier, some companies in this category will implement solutions incorporating some form of restriction on their code, mainly by keeping a small section of the code closed, which will form the basis of their differentiation.

3.2. Services associated with free software

Considering the services associated with free software, there are many possible businesses because, in general, any services model based on proprietary software (such as those discussed in module 3) can be extrapolated to free software in a fairly direct way. All of the steps described in the chain of creating and implementing a technology solution are viable in the context of open applications. However, the use of free software extends the possibilities and differentiation factors of business models focused on services.

One of its basic differentiating principles is the **absence of licensing costs**, giving it a clear competitive advantage over proprietary solutions. Nonetheless, in order to take advantage of this factor, it is important for the proposed solution to be cheaper in the long run (considering the "total cost of ownership") and to provide a standard of quality at least equivalent to its proprietary competitors. It is also crucial for companies offering free software services to be more attractive to customers by reducing the possibility of lock-in situations: these providers cannot rely on continued income in a situation with captive customers; instead, they must be based on the **continued provision of quality services**.

On the other hand, just because a software is free, this does not mean that it will be accessible to everybody. The market for service companies will not diminish due to the availability of free applications or those at no cost, since the task of selection, installation, training and support will always be necessary in corporate environments, and it will be more interesting if the licensing budget is spent on improving service.

As a rule, these types of company are involved in various projects, though not intensively in any. Some will contribute, as is the case of platform distributors, with debugging, especially in areas of customer interest, and on the tasks of integration and ensuring compatibility between different applications. Others, such as those that focus on consulting and selection (with no capacity for development), will not contribute to the projects on which they are based, since their work is usually kept private and will not be visible to the public. In these cases, however, a return can be obtained in the form of the promotion and adoption of the solution on which they work.

There is a vast range of possible models in this category (differentiation with respect to size, solution segmentation – horizontal or vertical – industry segmentation, specialisation in a particular service: custom development, selection, consulting, integration, training, etc.), and most companies will offer a combination of the possible services. First of all, we will look at the special features of free software in the different stages of implementation of a technology solution, before turning to the specific typologies of business models, which group certain services in a particular way.

Custom developments

Free software offers companies a compromise on the question of "**to buy or to develop**". These companies can start with a free standard product and, either internally or through a development company, build the necessary adaptations to suit their needs. Both the service companies that we will look at now and the product-oriented ones we saw previously will receive offers to perform this type of customisation. However, making these adaptations privately, without trying to incorporate them into the master project, can be problematic when it comes to maintaining compatibility between the adaptations and subsequent versions. Hence, working with the community, designing the new features so that they can appeal to more people, and incorporating them into the main code of the project will save a lot of work and complications.

Selection

The presence of a wide range of applications within the (economic) scope of any company makes selection a critical task. Not only will it be necessary to find products that better suit the needs of the client company, they must also evaluate the health of certain projects, the pace of debugging and new releases, and their stability. For corporate environments, a project with a lot of movement and a rapid rate of adoption of improvements may not be the best, since a stable product that will not change significantly over time may be more appropriate.

Installation and integration

Although this phase also generates needs in commercial environments, free software has a special business opportunity in this field: its lack of packaging and final finish. In*Open Source for the Enterprise*, Woods and Guliani allude to the concept of "productisation" as one of the main shortcomings of free software for achieving widespread adoption. The term refers to the degree to which the application has been packaged and prepared for end users, with the development of automatic installers, graphical configuration interfaces and sufficiently detailed documentation which, in short, allow for its installation and use by inexperienced users.

Additional reading

D. Woods; G. Guliani (2005). *Open Source for the Enterprise: Managing Risks, Reaping Rewards*.

As a general rule, commercial software comes more packaged and finished than the free software developed on a voluntary basis. The installation scripts, administrative interfaces and documentation are usually more complete for a proprietary commercial product than for a free software product of the same age. While this lack of product completion is irrelevant for technology enthusiasts – indeed, it can even be more attractive because the adaptation and administration can be more direct and personal – to cross the chasm and reach the corporate client, free software must have a higher degree of packing and finishing. According to Woods and Guliani:

> "A broad oversimplification about open source versus commercial software is that open source represents primarily an investment of time, and commercial software represents primarily an investment of money. Any organization setting out to use open source must set aside some time for research and experimentation. "
>
> Dan Woods and Gautam Guliani. "Open source for the enterprise"

This time investment for completing an open-source application or selection of applications offers an important business opportunity both for platform integrators and developers. Hence, a good symbiosis could be established between the private sector and non-profit free software projects in which the investment would be spent on more monotonous work, leaving the more creative and innovative work to the volunteer community while also allowing the simultaneous creation of more mature products that are more likely to attain a high level of adoption.

Furthermore, both the modularity of free software and its coexistence with proprietary systems can generate serious compatibility problems, which require painstaking integration. The generalisation of standards will be beneficial for minimising the adverse effects of combining different software elements.

Technical certification

The inherent features of the finish of free software also allow for the possibility of certification by integrators and external consultants. This can take two forms: certification of compliance with international standards or certification of suitability for specific technology environments. The certifier provides assurance that the package meets a series of requirements and is legally responsible for their compliance.

Hence, the certifier provides an intermediary responsible for a set of solutions, an essential factor for many new technology departments of software consumer companies. Often, when an information technology department arranges support and maintenance, it is not only hiring a method of resolving incidents, but also a person or company to which it can attribute the problems or failures that may arise. The decision to adopt a particular free software

Additional reading

S. Sieber; J. Valor (2005). *Criterios de adopción de las tecnologías de información y comunicación*. IESE.
<www.iese.edu/en/files/6_15211.pdf>

solution without intermediaries to offer guarantees puts all of the burden of success or failure on the department itself, which may prefer for the intermediary to assume this responsibility.

Training

Training can be a source of easy income. In addition to the fact that the open development model makes the information on a product available to everybody, most free software projects lack formal training programs, meaning that anyone can enter the business. Many established companies whose business is training have added free software programs to their offer.

Support and maintenance

We have already seen how support and maintenance services are an important source of revenue for companies engaged in the development of a free product, but they also form part of the offer of companies that only provide horizontal services, as we shall see below.

As we said earlier, the possible range of service companies is vast, with models being developed on the basis of specialisation in certain services, a type of applications, local market or large scale, etc. We will study three typologies in detail. Firstly, platform distributors, as they were one of the first business models implemented with free software and are fairly representative of a number of major companies in the sector. Secondly, we have chosen two examples at either end of the scale: large integrators and small niche micro-enterprises. Between the two, we have the other possible business models, which focus on the provision of services.

3.2.1. Platform distribution companies

The activity of this type of company is concentrated on the integration and selection of components to generate a **comprehensive software solution**. The diversity of applications and results generated by the free software development model requires integrated teams to give cohesion and ensure the compatibility of the parts. This has given rise to the emergence of different distributions developed by different actors. This activity is also an obvious potential business model.

Platform distribution companies use a similar model to application development companies and service providers, but the **selection and integration of a broad product base**, as opposed to development, lies at the crux of its work.

Companies using this model generate and distribute integrated software packages, mainly for corporate customers. The platform generated is the company's core product, which generates one major problem: product differentiation is very difficult because it is freely accessible.

Besides distributing software under a traditional model through the sale of packaged CDs, these companies often supplement their offer with services such as installation and quality support, often through a subscription system. Their added-value is based on reliability and trust, conveyed by the brand that represents them. They offer to fill in the gaps that a free software product may have for corporate environments, which seek an appropriate, stable and reliable solution – even at the cost of features and performance.

Thus, their potential customers will be medium and large enterprises, which require maturity and stability, professional support and a viable ecosystem of solutions. The investment in software is amortised over five years, so a company that is going to invest in software will need to know that – at least for this period – it will have support for these products. Given the extra costs associated with switching from one technology solution to another, having support that lasts beyond the amortisation period is highly desirable.

Hence, generating trust is fundamental to their business strategy and must include the development of a brand that conveys added reliability to a free software product. Given that their business model is based on a product freely accessible to anyone, these companies seek to develop a strong brand as a differentiating factor that will allow them to gain market shares over the same or very similar products.

Although these companies do not usually focus on the development of specific applications, they do often contribute to projects that they draw on by debugging, and develop new products only when necessary to expand the market for their product.

Red Hat

The archetypal example of a platform distributor is Red Hat, Inc., and this is also the model followed by Novell with SUSE, Canonical with Ubuntu, and Caldera Systems with Caldera Linux.

New distributors

New distribution companies have recently emerged, offering more specialised software bundles for more limited markets. SourceLabs, SpikeSource and Wild Open Source are examples of such initiatives. SourceLabs, for example, offers certified collections of software usually used together, such as Linux, Apache, PHP and MySQL. Wild Open Source, on the other hand, customises distributions for use in high-performance contexts or embedded systems. Along with the certified bundle, the companies offer maintenance and support services for their selection, just like traditional subscription companies.

The main challenge for this type of company will be to define software collections that are wide-ranging enough to maintain a sufficient customer base while being able to provide support for all elements in the bundle.

The SpikeSource case

Company name	Spikesource, Inc.
Head office	Redwood City, CA. (United States)
Website	www.spikesource.com
Creation date	2003
No. of people employed in 2006	80
Turnover in 2007 (million)	

Corporate data on SpikeSource, Inc. Prepared with statistics from Hoovers (http://www.hoovers.com)

SpikeSource is a representative example of the business potential generated by the lack of finish of free software products. Set up in 2003 by one of the most important venture capital firms of the Internet boom – Kleiner Perkins Caufield & Byers – SpikeSource launched its first products in April 2005. In October 2006, the company announced its expansion into Europe through a network of local solution providers and technology partners.

Murugan Pal, founder, summarises the company's activity as follows:

SpikeSource's goal is to facilitate the adoption of open source software in the enterprise through testing, certification and support services. We innovate, automate and optimize advanced testing techniques as part of our core competency."

(Murugan Pal. "Participatory Testing: The SpikeSource Approach". http://www.oreillynet.com/pub/a/network/2005/04/07/spikesource.html)

As differentiating factors with classical integrated solution distributors like Red Hat, the company highlights its efforts to promote testing automation and its combination of specific applications that can be installed on different platforms and operating systems. It includes versions for different operating systems, both free and proprietary, and incorporates closed software in some products.

In addition to its bundles, such as SpikeWAMP-1.4, which includes the latest versions of PHP, MySQL and Apache (for Windows installation), and "SuiteTwo", which integrates a wide range of embedded collaborative applications, and "Web 2.0" features, it recently launched a platform for developers on which they can test and integrate their applications, thus obtaining SpikeSource certification and a better software finish (productisation) as a result.

The work of this type of company can be very positive in increasing visibility and promoting the adoption of free software solutions, and SpikeSource has tapped into this. The company makes great efforts to show that its work benefits the free software community – and that it does not simply exploit it – by including well-known figures from the world of free software, such as Brian Behlendorf and Larry Rosen, on its advisory committee as endorsements. It has also developed a website for developers (http://developer.spikesource.com), where it offers its automated testing services for integration and compatibility on various platforms.

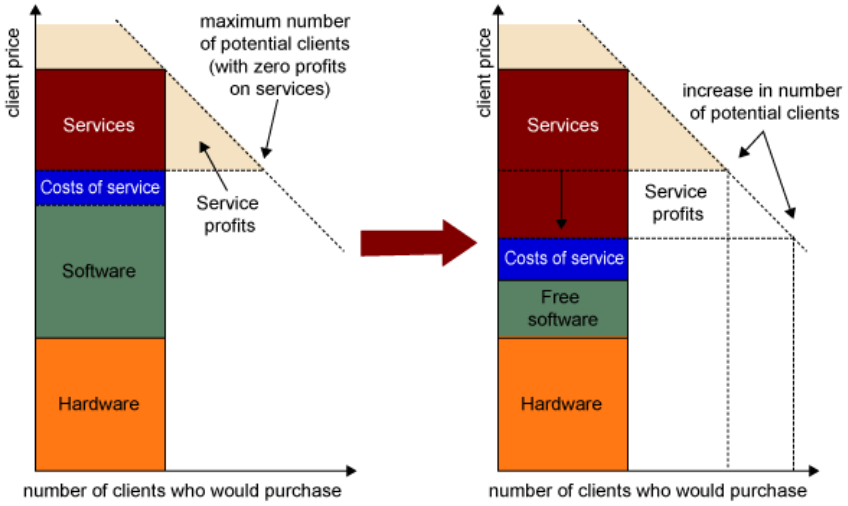
Nonetheless, the **automation software** used by the company combines parts that have been released with parts that remain closed. In this case, reserving a portion of the code is a strategy to protect its differentiation and keep competition from comparable services at bay. This decision reveals that rather than losing revenue from licensing (which, as we have repeatedly seen in this subject is not a real obstacle), the use of free software affects the company's possibilities of differentiation – and hence, business. In the case of SpikeSource, the effort invested in its testing applications will be rewarded not by the sale of licences for this software but by the protection of its differentiating factor from other companies offering similar services.

3.2.2. Large integrators

Large systems integrators or solution generators are one of the types of company that stand to gain the most by basing their business on free software, given the direct cost savings, and the subsequent possibility of reaching more customers.

Clients usually look for companies that can provide solutions to an information and communication technology (ICT) problem and are not concerned with implementation details. A complete solution will combine hardware, software and services, making the process easier for the customer, who need only contact a company to solve its ICT problems and not have to worry about compatibility between providers. Therefore, everything that the company can save on software costs by using free software can be transferred to the costs of services, which will enhance the solution. The company can slash prices to increase its potential number of customers, or simply enhance its profitability. This type of large integrator, which generally works on complex projects, can maintain its prices due to the barriers to the entry of other competitors.

The figure below illustrates this situation, outlining the demand curve for comprehensive solutions and provider costs.



Demand curve of comprehensive computer services. Sales margins and number of clients. Source: Dirk Riehle, "The Economic Motivation of Open Source Software: Stakeholder perspectives". http://www.riehle.org/computer-science/research/2007/computer-2007-article.html)

There are many consulting and selection firms, including Ayamon, Enomaly, Navica, OpenLogic, Optaros and X-tend. Large integrators include IBM, Sun and HP.

The IBM case

Company name	IBM

Corporate data on IBM. Prepared with statistics from SoftwareMagazine (www.softwaremag.com) and Wikipedia.

Head office	Armonk, NY (United States)
Website	www.ibm.com
Creation date	Its origins date back to 1896. In 1924, it changed its name to IBM
No. of people employed in 2007	394,540
Turnover in 2007 (million)	$91,423

Corporate data on IBM. Prepared with statistics from SoftwareMagazine (www.softwaremag.com) and Wikipedia.

Twenty years ago, **IBM**was one of the strongest advocates of intellectual property rights for software. Its argument was that without strong copyright protection, there would be no incentives for companies to invest in software development.

Now, although it has retained the bulk of its proprietary software, IBM has launched major campaigns in support of free software, offering considerable financial contributions to the development of Linux and other applications, and the release of applications such as the Eclipse development platform and part of its AIX operating system.

IBM's current business model focuses on the **sale of high-end hardware**, proprietary software on Linux and the provision of **integration services for corporate clients**. Although IBM has been one of the world's leading software manufacturers, its programs have usually been marketed as a combined solution with its own hardware. As a result, the company has little to lose from lack of differentiation in the software that it uses: given the barriers to competitor entry in mainframes, the use of low-cost software allows the company to cut its prices and extend its range of customers without undergoing a loss of differentiation that would significantly increase its competition.

Thus, its use of Linux allows IBM to offer a lower overall price for its hardware and services, while also providing a common platform on which to build and sell applications and special services. Along these lines, we can also mention the savings made by the company through the use of an operating system with wide prior adoption – in marketing, distribution and sales terms – as well as the reduction in risk and investment in development. Moreover, the public image benefits obtained have also been significant.

Naturally, IBM's free software activity involves a more complex strategy that affords it a better competitive position on several fronts. From strategies based on the use of free software to enhance the marketing of its proprietary products (such as "loss leaders" and free kernel plus proprietary accessories) to gaining a better position than other big software providers.

The use of free software has given IBM more independence than other large companies, such as Microsoft, and a better position over direct competitors like Sun. The latter has, for a long time, based its business strategy on the combined sale of hardware plus "better than average" operating systems and would therefore have more to lose in the event of cost-cutting and the presence of equivalent low-cost software.

3.2.3. Software services: small and micro-enterprises

Another basic phenomenon sparked by free software is the **transfer of knowledge and technology**. Investments in training, development and technology, both on the scale of large companies and at individual level, is available through developments that are open to anyone with an Internet connection and a certain knowledge.

This phenomenon can have a major impact on technology transfer between countries that are more or less developed, and internally, between large multinationals and local micro-enterprises.

The possibility of free access to both the code and decisions on design and development offers great potential to small technology companies, which can be in contact with and adopt the most innovative technology backed by large financial investments.

Given their size, these companies generally base their activities on specific niches and require only a few customers to stay in business. The possibilities of market segmentation are endless, but one common factor is that of closer and more personalised attention (many customers prefer to be the big customer of a small business than a small customer of a large multinational).

The more relevant companies of this nature include those that base their differentiation on the use of free software not only for the benefits we have mentioned thus far, but as a **statement of intent**, as yet another element of a business logic that seeks not to accumulate profit but to generate self-sustaining livelihoods through the provision of services that contribute to the development and well-being of society.

The inner workings of these companies also often reflect this philosophy and approach to business, based on **horizontality** and **transparency**. Interestingly, Spain's legal framework provides for a concept of business substantially aligned with what we have described: worker cooperatives, in which there are no financial backers and the workers themselves manage and control the company.

Again, the concept of **business ethics** is neither new nor unique to free software but takes on a special meaning in this type of company. Often, these small businesses form groups through different types of networks, which is a key strategy for encouraging support and cooperation between them, in line with the ethical and political principles on which they are based.

A considerable proportion of the potential customers are other companies with similar operating principles, organisations with social or political motivations, and government bodies.

Examples of this model include several Spanish companies with a similar type of operation, which have been uniting in the Ikusnet group (http://www.grupoikusnet.com/) under the following principles:

> "Our methodology is based on cooperation and 'horizontality' in making and implementing decisions, to the extent that the mode of cooperation itself becomes a productive force that seeks to deliver its effects in the framework of the information and knowledge society."

We can also mention the Madrid-based cooperative **Xsto.info** (http://xsto.info), a micro-enterprise with less than ten workers. Born at the heart of social movements, it was established as a worker cooperative in 2003. This choice of legal form is, like the use of free software, a statement of intent regarding its operating principles, which are complemented by the website, in line with the following motto:

> "There is still time to take part in this social transformation to ensure that it occurs in a participatory, open, free and democratic way".

Among its customers we find local authorities such as Parla City Council, and various types of association, including the Federación Regional de Asociaciones de Vecinos de Madrid (Regional Federation of Neighbourhood Associations of Madrid).

Another very representative example, particularly interesting given its age, is Easter-eggs, which we will now discuss in detail.

The Easter-eggs case

Company name	Easter-eggs
Head office	Paris, France
Website	www.easter-eggs.com
Creation date	1997
No. of people employed in 2007	15
Turnover in 2006	€800,000

Corporate data on Easter-eggs. Taken from its website (http://www.easter-eggs.com)

Easter-eggsis a French SME with a consolidated track record that provides services for free software. Founded in 1997, it offers a wide range of services, from the installation, administration and security of GNU/Linux systems to the adaptation of applications and custom developments and consulting, auditing and training. The company offers services for older free software – and still looks healthy: profitable from the moment it was created, it now employs fifteen people and obtained a turnover of €800 thousand in 2006. Its clients include the René Descartes University of Paris (http://www.univ-paris5.fr/) and Europcar, for which it implemented a GNU/Linux migration programme in 3,500 of its agencies.

For the company, the decision to provide services for free software was based on ethical rather than financial principles, and these principles are also what led it to define a very unique method of business operation. In a manner similar to that of the operation of Spanish worker cooperatives, Easter-eggs is fully and solely controlled by its employees. There are no venture capitalists or foreign investment of any sort. An association was set up to implement this organisational system, the Association of Easter-eggs Employees (http://www.easter-eggs.org), which holds a 99.8% stake in the company.

These were the foundations on which Easter-eggs built its business differentiation, defining itself as a "social company" with a central concern for creating a "citizen-based company" that responds to the growing aspirations of citizens who are beginning to realise the limits of consumerism and demand that companies act with purpose. Its operating principles include financial transparency (its accounting records are available for download from its website: http://www.easter-eggs.org/rubrique_10_Comptabilite.html), equal pay and mechanisms for the involvement and co-responsibility of its employees.

As part of its strategy to create networks and bring together small, socially-responsible businesses to provide services on a larger scale and as a method of joint promotion, in 2002, the Easter-eggs association created the *libre enterprise* network (http://www.libre-entreprise.org), which encompasses approximately sixteen French companies offering free software-based services, all with similar business models.

3.3. Ancillary markets: hardware

One of the first business models described by Hecker, "Widget Frosting" is still as valid today as it was then. For hardware manufacturers, the development of software is a necessary expense if they are to sell their products, so any strategy that will lower the associated costs is desirable. In addition, following a model of free software development extends the possibilities of portability to other platforms, thereby increasing the market segment. We saw earlier how the major providers, which include hardware in their offer, are incorporating free operating systems as a way of reducing the final costs of the service, thus increasing their potential customer base.

On this point, it is interesting to note the role that Linux is playing in the new generation of **embedded devices**. We are witnessing a return to the combined sale of hardware and software in this type of device, which must come with its specific functionality built-in, often with simple operating systems with limited functionality. Nonetheless, the possibility of using embedded Linux has increased the business opportunities for this type of hardware.

The use of free software offers significant advantages in terms of cost savings, shorter development periods (essential in a market governed by short life cycles), ease of development subcontracting (due to a highly modular existing base) and the possibilities for innovation introduced by setting up a community around the product. Moreover, the use of free software gives manufacturers significant independence from the Windows Mobile and Symbian platforms, and hence, from the agendas of Microsoft and Nokia.

Currently, Linux-based operating systems are the most common in embedded systems and their adoption by consolidated companies of the sector, such as Wind River, points toward the continuation of this trend. In the smartphone market, Linux increased from 3.4% in 2004 to 14.3% in 2005, while embedded Windows only grew from 2.9% to 4.5% in the same period.

Recommended website

For more information:
Alejandro Lucero, "Seminario UAM: Linux en Sistemas Empotrados". *www.os3sl.com/Documents/Seminario_UAM_I.pdf*.

Furthermore, the existence of software at an affordable price for a large audience also generates an ecosystem of needs around it, which the hardware often forms part of. The **Asterisk** IP voice platform, for example, allows many businesses to use switchboards, with a significant reduction in costs. However, it requires users to purchase certain hardware elements, such as IP terminals, Asterisk cards, routers, recording systems, etc.

The manufacturers of these products can benefit from the spread of software like Asterisk, so they will have much to gain from participating in and contributing to its development. Likewise, software development companies can earn money by selling hardware and related services, as is the case of Digium, the company chiefly responsible for the development of Asterisk.

There are also other spaces and niches that can be exploited through this technology, such as those tapped by **Avanzada7**. This Málaga-based company sells the necessary hardware for the implementation of Asterisk, but acknowledges that it is neither a manufacturer nor a major distributor. Its differentiation stems from the provision of free support services following the sale of the devices. Avanzada7 has also established a partnership with Digium, the company responsible for the development of the software, creating a trusted network that extends to other companies wishing to implement Asterisk for end customers. Thus, it has set up a pyramidal network of the type described above based on the needs generated by free software, which it exploits through *coopetition* strategies.

The Chumby case

Company name	Chumby Industries, Inc.
Head office	San Diego, CA (United States)
Website	www.chumby.com
Creation date	2005
No. of people employed in 2007	
Turnover in 2006	

Corporate data on Chumby Industries, Inc.

Chumby Industries was set up with the aim of creating and marketing the "Chumby", launched in August 2006. This wireless (Wi-Fi) device was designed to replace the clock radio and can connect to the "Chumby Network", where it can download different types of information. It can play podcasts, Internet radio, and some videos. The device runs Linux and Flash Lite, an Adobe program with small interactive applications or "widgets". It does not have a browser and contents can only be downloaded through widgets, each of which has its own specific function: read the latest news from a blog, download the latest photos from a gallery, etc.

The Chumby hardware and software are free and both its schemas and printed circuit boards – and even its source code – can be downloaded. The company's marketing activity is based on its openness: the Chumby can be customised at any level by changing the outer casing and (literally) sewing on extensions to taste, creating new widgets or hacking the hardware. Thus, the device is not only sold as "user-friendly", it also opens the door to the expansion of its features beyond the control and financing of the company, leaving it to evolve into what every user wants it to be.

Nonetheless, Chumby's business model is not aimed at obtaining revenue from hardware, and the price of the device is relatively low. Steve Tomlin, founder and CEO of the company argues that several business models were possible with Chumby: they could have charged more and followed the model of a traditional hardware vendor, with the problems of recurring revenue that this would generate, or they could charge little for the device, but then charge for content subscriptions. However, the company preferred

a third way: to obtain the revenue needed to just cover costs with sales and generate its profits through advertising.

To secure this new field of business, Chumby is not 100% open and there are constraints on its use, both in the hardware and on the "Chumby Network", thus guaranteeing the business model.

"Chumby network" access

After purchasing a Chumby, the user must register on the company website to access the widgets, accepting their terms of use. These terms allow anybody to add new widgets with any type of information they wish, giving their permission to distribute this information to any device connected to the network. However, restrictions are placed on permitted content, and inappropriate content (racist, violent, sexist, spam, etc.) is banned, as is commercial content:

"**Prohibited Content** includes Content that: (...) except as expressly approved by Chumby, involves commercial activities and/or promotions such as, without limitation, contests, sweepstakes, barter, advertising, or pyramid schemes." (http://www.chumby.com/pages/terms)

A payment must therefore be made to obtain authorisation for advertising content. The terms and conditions also warn that the user will receive advertising when he/she connects to the Chumby network.

Although widgets can technically be incorporated outside the Chumby network using USB devices, the company is confident that most of the contributions will remain within its network, thus attracting enough content to generate value from the number of people and contributions on it.

The device

Chumby allows access to the schemas and PCBs of its device. However, manufacturers seeking to use its designs and incorporate them into their own products have to pay the company to licence their new product. In addition, they have to accept that, besides any other networks to which they connect, they will also incorporate the Chumby Network.

To summarise, Chumby acknowledges that the value of its device lies in the content, in a manner similar way to O'Reilly in "Open Source Paradigm Shift" and others. Its strategy, besides characterising the product by its openness, is to attract as many people as possible to the network in an attempt to make it a benchmark network for small mobile devices of this nature. However, instead of selling content through subscriptions, it has decided to capitalise on this value through advertising.

For the company, the use of open hardware and software is a key strategy for the spread and adoption not just of its device but of the network that it has created to provide content. Moreover, its openness gives it a clear differentiation and commercial edge over similar products like Apple's iPod Touch and iPhone.

3.4. Other ancillary markets

The increased spread of free software, both due to its form of development and its use, generates other related markets that have been exploited by diverse companies:

- **Community and development:** perhaps the most obvious examples are those that provide hosting services and collaborative tools for software projects, such as SourceForge, CollabNet or Freshmeat. There has also been a proliferation of code search engines, including Google Code, Koders, Krugle and Codase.

- **Legal certification:** companies offering this type of certification are also becoming increasingly relevant. They ensure that a software or particular combination is legally possible and are familiar with the licensing prob-

lems it could have. This service is provided by the companies we saw earlier, such as the creators of platforms and bundles, like SpikeSource, but others have sprung up that focus entirely on legal issues, such as Black Duck and Palamida.

- **Sale of books:** O'Reilly and his books are one of the most often cited examples in this category.

- ***Merchandising:*** we should not overlook the importance of merchandising as a supplementary or even main form of financing. Examples include ThinkGeek, a subsidiary of SourceForge, which contributes revenue through Internet sales of various types of product for "geeks": from t-shirts and mugs to a range of gadgets.

Summary

This module has looked in detail at the diverse valid and viable business models based on free software. The growth of companies that focus entirely on its exploitation and the redirection of the strategy of software multinationals is conclusive proof.

Initially, we described different classifications proposed by a range of authors over time:

- The classifications of Hecker and Raymond, based on the observation of companies that used free software as part of their business models.

- The classification of the European Working Group on Libre Software, based on the business financing model.

- Daffara's classification, based on empirical studies.

Finally, we proposed and developed our own business models proposal:

- Specialist/vertical, focusing primarily on the free software product and which can adopt mixed dual licensing models, proprietary accessories, distributed product sales or service provision models for the product, such as software as a service.

- Associated services such as custom developments, product selection, installation, integration, technical certification, training, support and maintenance, which may be oriented towards the distribution of platforms, large scale integration or the service of small businesses and micro-enterprises.

- Ancillary hardware markets, which use free software to complement their main business: the sale of physical products or the business of contents accessible from a particular hardware.

- Other ancillary markets, such as collaborative tools, legal certification, the sale of books or merchandising.

Bibliography

Augustin, L. (2007). "A New Breed of P&L: the Open Source Business Financial Model". Open Source Business Conference (OSBC)*Metcalfe, Randy*. <http://www.osbc.com/live/images/13/presentation_dwn/A_New_Breed_of_P_and_L.pdf> [Consulted in February 2009]

Mickos, M. (2007). "Open Source: why freedom makes a better business model". *Open Source Business Conference (OSBC)*. <http://www.osbc.com/live/images/13/presentation_dwn/Keynote-Open_Source_Why_Freedom.pdf> [Consulted in June 2008]

West, J. and Gallagher, S. (2006). "Patterns of Open Innovation in Open Source Software". In: Henry Chesbrough; Wim Vanhaverbeke; Joel West (eds.). *Open Innovation: Researching a New Paradigm* (pp. 82-106). Oxford: Oxford University Press. <http://www.openinnovation.net/Book/NewParadigm/Chapters/index.html> [Consulted in June 2008].

Capiobanco, F.; Onetti, A. (July 2005). "Open Source and Business Model Innovation. The Funambol case". In: M. Scotto; G. Succi (eds.). *Proceedings of First International Conference on Open Source* (OSS2005) (pp. 224-227). Genoa. <http://oss2005.case.unibz.it/Papers/4.pdf> [Consulted in June 2008].

Riehle, D. (2007). "The Economic Motivation of Open Source Software: Stakeholder perspectives". *IEEE Computer* (vol. 4, no. 40, pp. 25-32). <http://www.riehle.org/computer-science/research/2007/computer-2007-article.html> [Consulted in February 2009]

Comino, S.; Manetti, F. M. (2007). *Dual licensing in open source markets*. Università Degli Studi di Trento, Department of Economics. <http://www-econo.economia.unitn.it/new/pubblicazioni/papers/18_07_comino.pdf> [Consulted in June 2008].

Daffara, C. (2007). *Business models in FLOSS-based companies*. Conecta Research, 2007. <http://opensource.mit.edu/papers/OSSEMP07-daffara.pdf> [Consulted in June 2008]

Pal, M. (July 2005). "Participatory Testing: The SpikeSource Approach". O'Reilly Network. <http://www.oreillynet.com/pub/a/network/2005/04/07/spikesource.html> [Consulted in June 2008]

Kelsey, J.; Schneier, B. (June 1999). "The Street Performer Protocol and Digital Copyrights". *First Monday* (vol. 4, no. 6). <http://www.firstmonday.dk/issues/issue4_6/kelsey/> [Consulted in June 2008]

Rasch, C. (June 2001). "The Wall Street Performer Protocol". *First Monday* (vol. 6, no. 6). <http://www.firstmonday.org/issues/issue6_6/rasch/index.html> [Consulted in June 2008]

Daffara, C.; Barahona, J. B. et al (2000). "Free Software/Open Source: Information Society Opportunities for Europe?" *Working paper*. <http://eu.conecta.it/paper/> [Consulted in February 2009]

Lucero, Alejandro. "Seminario UAM: Linux en Sistemas Empotrados". <www.os3sl.com/Documents/Seminario_UAM_I.pdf> [Consulted in June 2008]

Raymond, E. (1999). *The Magic Cauldron* <http://catb.org/~esr/writings/magic-cauldron/>

Spanish translation in: <http://gnuwin.epfl.ch/articles/es/magiccauldron/es-magic-cauldron/es-magic-cauldron.html> [Consulted in February 2009]

Hecker, F. (1998). *Setting Up Shop. The Business of Open Source Business* <http://hecker.org/writings/setting-up-shop> [Consulted in February 2009]

Metcalfe, Randy (2006). *Open Source Business: Differentiation and Success* <http://www.osswatch.ac.uk/resources/businessmodels.xml> [Consulted in February 2009]

Case studies

50 Open Source Success Stories in Business, Education, and Government <http://www.blogcrm.com/50-open-source-success-stories-in-business-education-and-government.php>

Red Hat and J. Boss. "Is Open Source viable in Industry? The case of Red Hat and JBoss". <http://www.whyfloss.com/es/conference/madrid08/getpdf/68>

Avanzada7. "Business models based on Asterisk: The case of Avanzada7". <http://www.whyfloss.com/es/conference/madrid08/getpdf/64>

Openbravo. "Openbravo: keys to success in free software application development". <http://www.whyfloss.com/es/conference/madrid08/getpdf/49>

Liferay. "Liferay Enterprise Portal: The project, the product, the community and how to extend it". <http://www.whyfloss.com/es/conference/madrid08/getpdf/66>

Various cases. <http://www.opensourceacademy.gov.uk/solutions/casestudies>