Software as a business

Irene Fernández Monsalve

PID_00145051

Universitat Oberta de Catalunya
www.uoc.edu

Index

Introduction

In this module, we will look at the "classical" view of software as a business. We will focus on proprietary software, leaving the study of the further possibilities of free software in this scenario for a later module. Although some of the aspects that we touch upon will be irrelevant when it comes to the application of free software strategies, others will still be valid to a large extent.

We will review some of the key factors to consider when designing a business around software, such as the **choice of the main activity** and the general approach of the company (selling products or services), aspects of **selling** and **marketing** (how to choose our market and how to approach it), and the **definition of its products or services** (what type of products or services to develop and how to position them).

Objectives

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

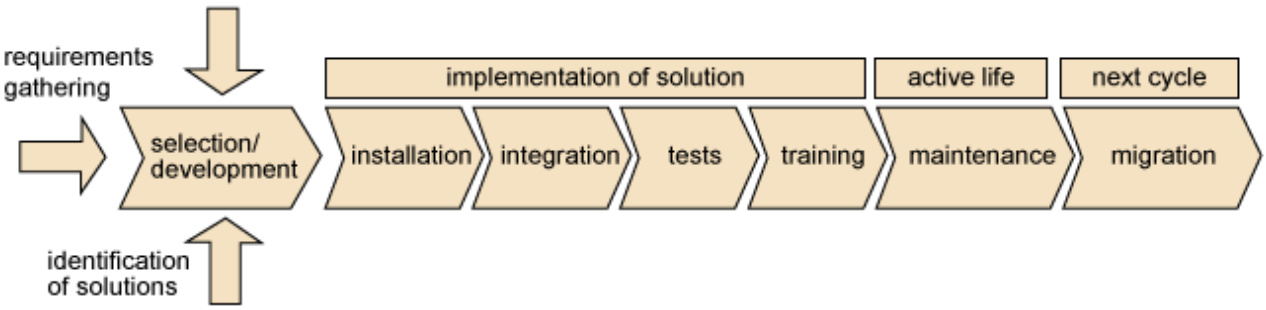
1. To obtain a global vision of the business opportunities of software.
2. To learn about the traditional models of software companies.
3. To understand the economic features of and differences between product companies and service companies.
4. To identify the key factors that software companies need to consider when positioning their products on the market.

1. Business opportunities with software

Both individuals and corporate environments have software needs that generate multiple business opportunities.

The basic task involved in meeting these needs is to create this software, the task of development per se. However, the needs to be met do not end here; this is only the beginning. Once the product is made available, a number of related needs arise, such as consulting, installation, configuration, maintenance, support and training, for which certain customers (mainly other companies) are willing to pay.

Throughout the process of technology adoption, from the identification of needs to the decision to build or buy, right up to the end of the useful life of the technology, multiple needs are generated that can be met by many different companies:



Process of technology adoption *(based on Carlo Daffara. "Sustainability of FLOSS-Based economic models". II Open Source World Conference.* Málaga. Available at: http://www.cospa-project.org/Assets/resources/daffara-OSWC2.pdf)

Moreover, the process of software creation itself can be interpreted in two ways: as the creation of a product or as the provision of a service. The choice between the two will be critical for defining the company's operation and its potential generation of revenue, which will result in very different business models.

This choice – developing software as a product or a service – also reflects the first issue that a company that consumes software will need to evaluate when adopting a technological solution: whether to purchase a standard, packaged product or to obtain a tailored development.

We can therefore distinguish between the following business activities in relation to software:

- Application development
  - As a product: standard solutions (shrink-wrapped)

  - As a service: custom development

- Provision of services around one or more applications
  - Consulting
  - Selection
  - Installation
  - Integration
  - Training
  - Maintenance and support
  - etc.

- Software as a service

This classification is intended to be neither exhaustive nor exclusive, that is, many companies will implement hybrid models allowing them to provide integral solutions to their customers.

The features of software companies and their business dynamics will vary greatly depending on the activities that they focus on, as we shall see later, but any of the models has the potential to generate both viable and highly profitable businesses.

1.1. Service companies

As we explained earlier, companies can specialise in one or more aspects of the chain of technology adoption and implement a number of activities at the same time.

Hence, for companies that include various services in their business offer, we can distinguish between two types of specialisation: vertical and horizontal.

1.1.1. Vertical specialisation

Broadly speaking, companies whose main activity is development will tend to have a **vertical specialisation**. If there business strategy is centred on custom development, their activities will naturally include other related services, such as **installation**, **integration** and **training**. However, as we shall see, companies that adopt the strategy of software as a product will also do well to exploit associated services as a way to guarantee a steady flow of income.

	Package 1	**Package 2**	Package 3	etc.
Development	X	X		

Vertical specialisation (based on Daffara. "Sustainability of FLOSS-Based economic models". *II Open Source World Conference.* Málaga. Available at: http://www.cospa-project.org/Assets/resources/daffara-OSWC2.pdf)

	Package 1	**Package 2**	Package 3	etc.
Installation	X	X		
Integration	X	X		
Certification	X	X		
Training	X	X		
Maintenance and support	X	X		
Migration	X	X		

Vertical specialisation (based on Daffara. "Sustainability of FLOSS-Based economic models". *II Open Source World Conference.* Málaga. Available at: http://www.cospa-project.org/Assets/resources/daffara-OSWC2.pdf)

Interestingly, a company that invests a certain amount of money in software licensing expects to invest additional sums in related services, such as maintenance and support, and in updates. Thus, selling products to business clients will open the door to obtaining service contracts with the same clients and hence, a more consistent flow of income over time.

1.1.2. Horizontal specialisation

In contrast, companies that exploit the needs generated by the general use of software products will often offer services in a variety of packages, focusing on one or more of the phases of the adoption of a technology.

	Package 1	**Package 2**	**Package 3**	**etc.**
Selection/Custom developments				
Installation				
Integration				
Certification	X	X	X	X
Training	X	X	X	X
Maintenance and support				
Migration				

Horizontal specialisation (based on Daffara. "Sustainability of FLOSS-Based economic models". *II Open Source World Conference.* Málaga. Available at: http://www.cospa-project.org/Assets/resources/daffara-OSWC2.pdf).

Although some companies specialise in training or support, service companies often touch on a number of the phases described, generating typologies such as consulting (with an emphasis on selection, advice, and/or certification), or integral solution providers, which cover all categories, including custom developments and even the provision of hardware.

Companies that create GNU/Linux distributions use a **service provision model with horizontal specialisation**.

Example of horizontal specialisation

Canonical, creators of the distribution based on **Debian Ubuntu**, perform a task of selection and horizontal integration that encompasses a complete operating system along with several applications, with the basic aim of providing a distribution that is easy to use, install and set up under the slogan "linux for humans". However, since Ubuntu is free software, Canonical's income comes from related services, namely support, training and certification.

These service-oriented companies often observe that their clients prefer to receive integral solutions and deal with a single technology solutions provider. To be able to offer this comprehensive type of service, companies often need a powerful infrastructure and technical capacity, which limits the entry of SMEs as they are unable to meet every single need by themselves.

A common solution is for the service company to contract out the parts that it cannot handle alone. Another very interesting solution is the "pyramidal model of consulting" proposed by Daffara (*Sustainability of FLOSS-Based economic models*), which we will now explain.

Recommended website

For more information about the "pyramidal model of consulting", see: http://www.cospa-project.org/Assets/resources/daffara-OSWC2.pdf)

Generally speaking, computer support and maintenance can be said to follow the 80/20 rule: 80% of queries are easy and can be resolved immediately. The remaining 20%, however, are important problems and account for 80% of the effort. Hence, a service SME could take care of a high number of clients, dealing with 80% of their incidences and earning a reasonable amount for the service. To solve the remaining 20%, it will require the technical services of the software creation companies, who will obviously need to be paid more than what the company receives from each client but less than what it earns from all of these clients together.

This model will generate sustainable cooperation between the development companies, with vertical specialisation, and the companies offering integral solutions. The former will be able to reach more users through the horizontal consulting firms, which will also mean a significant source of income. The latter will be able to manage a large customer base and provide quality support for a range of products, maintaining a profitable business so long as the customer base is big enough.

1.2. Development companies: to create products or to provide services?

As we said earlier, a company that hopes to focus on development will have two main options to choose from: it could generate **standard products**, packaged to sell to the mass market (**shrink-wrapped**, as they are called), or it could generate **custom developments**, tailored to the needs of individual clients.

The first option has the potential of generating large profit margins but they will be difficult to maintain over time, and it has barriers to entry that could prove unsurmountable. The latter is a far more labour-intensive option with much lower profit margins, but it offers more possibilities of generating constant sources of income over time and of being less sensitive to changes in the macroeconomic environment.

We will now describe the differentiating features of these two options in detail.

Economies of scale and the possibility of large profit margins

Required reading

M. Cusumano (2004). *The Business of Software* (Chapter 1, "The Business of Software, a Personal View").

The economic process of software creation has special features not seen in other industries, affording it huge **positive returns to scale**.

On the one hand, commercial companies need to invest large sums of money in development before they can create a commercial version of a product to release, and they must often invest again every two or three years to maintain a constant flow of income. Since the aim of this development is to generate a standard product, it is very risky because there is no certainty that the investment will be recovered later through sales. However, once they have a finished product, the marginal cost of each additional copy sold is next to nothing. The first copy of the software created is very expensive, but the rest costs virtually nothing.

This leads to huge economies of scale on the supply side, which combine with significant economies of scale on the demand side: both due to the time invested in acquiring the skills to use an application and the possible incompatibility of formats, switching from one product to another is a difficult and expensive task. As a result, the bigger the user base of a product, the easier it is for this base to grow and survive over time. In the software market, then, we can come across "winner takes it all situations", where huge profits are generated and the entry of new companies to these markets is simultaneously blocked.

Examples of companies that generate standard products

The companies that have exploited these large economies of scale include some of the giants of the software industry, such as **Microsoft**, which tops the desktop operating systems market, and **Oracle**, with its purchase of **PeopleSoft** in 2005. However, there are small companies too, known as independent software vendors (ISV), that produce feasible businesses by exploiting specific niches. Examples include Pretty Good Solitaire, developed by the two-staff micro-enterprise **Goodsol Development Inc.** (one of the most popular solitaire games), and HomeSite, a HTML editor developed by the Bradbury Software micro-enterprise in 1995, which was purchased by Allaire Corp. (Allaire was later purchased by Macromedia, which, in turn, was absorbed in 2005 by Adobe).

In contrast, a company that engages in custom development will not have access to the economies of scale of standard software. Every new customer will require a specific development, making it a costly investment in time and effort, although this type of company does tend to reuse its developments where possible.

1.2.1. Need for initial investment

When we set up a company based on the traditional product idea, we come across an important problem: the **need for initial investment**. During the early stages of the company, dedicated to development, there will be no income flow, but there will be expenses until the first versions of the software are ready for release. Besides the expenses deriving directly from development,

we need to take into account the necessary expenses of marketing and sales. There are two solutions to this problem: **obtain external investment**, **or start another type of business activity** that generates sufficient income to allow for simultaneous development of the product.

Custom development companies entail **much less risk** and can start their activity with a much smaller investment (development only begins once a contract is signed), thus avoiding the need to search for outside investors.

The financial literature tends to focus on the discussion of companies that finance their development from venture capital investments since they are more attractive. This type of financing allows for faster growth, which is an important factor in success according to Cusumano. (Michael Cusumano, *The Business of Software*)

Reflection

At this point, we can consider the following: what parameters do we use to judge the success of a business initiative? Investors and financial publications consider a successful company to be one that manages to make a profit every year, and probably those that display growth too. A company that remains the same size with an income statement showing no profit will not attract the attention of investors or the financial literature. However, a company of this nature may have been very successful in creating quality jobs and maintaining them over time. For many entrepreneurs, this can be the main aim.

Obtaining sufficient outside investment can be an insurmountable obstacle and, even when it is possible, it has certain disadvantages that we need to take into account. The presence of investors will put pressure on the management decisions of the company, as it will have to generate sufficient profits to repay the investment and make gains. This situation will limit the autonomy and decision-making capacity of its founders.

The other option is not straightforward either. The company would have to redirect its business to services in an attempt to generate sufficient revenue from them to allow for the simultaneous development of the product. As we shall see later, it is difficult to be successful in this through the provision of services because the profit margin is smaller. Moreover, the lack of economies of scale and the presence of competition restrict the possibility of keeping prices high enough.

In this context, free software emerges with new features to alter the scenario. The possibility of **cutting costs** through the collaboration of volunteers, together with the new schemas of distribution and marketing offered by this collaboration constitute a relevant disruption of these scenarios and have the potential to significantly reduce the initial investment required.

We will look at these aspects in more detail in the following modules.

1.2.2. Maintaining the revenue stream

One of the basic questions that any company needs to ask is not only how to raise revenue at a given moment, but also how to maintain it over time. While continuity will be the norm for companies that focus on providing services (generally, if clients are satisfied, they will continue to need the services), in companies that focus on the production of standard solutions, the maintenance of a steady stream of income will be fraught by a range of problems.

1) Software cycles

Cusumano, in *The Business of Software*, compares the process of writing a successful software product to writing a best-seller. Doing so will generate huge profits but it is also very difficult and only occasionally generates the latter. The natural life cycle of a commercial software product will eventually cause it to lose the ability to generate income.

Initially, early versions will have several flaws and their functionality will not be finely tuned to the needs of users. This will allow the company that created it to maintain its income over time with the launch of new versions that gradually incorporate improvements into the product, both through debugging and by obtaining much more information on requirements from user and customer feedback.

Predictably, if new versions of the product contain sufficient improvements and are more attractive than the previous ones, they will continue to generate revenue. However, once users decide that the application is good enough, their motivation to pay for a new version will wane. Similarly, trying to maintain the income obtained from a best-seller with sequels has only limited effectiveness.

There are strategies to combat these trends and maintain a stream of revenue through the licensing of successive versions, typically at the expense of consumers. The total or partial incompatibility between successive versions of the product, coupled with intensive campaigns to publicise it, will lead to a new situation of economies of scale on the demand side in favour of the latest version, which will force many users to change even though the previous product met their needs.

However, due to the nature of some software products, constant updating is necessary due to changing user needs.

An illustrative example: accounting, labour and tax management applications.

Tax and labour legislation changes often, which means that users need to update their application every time this occurs. As a result, revenue can be kept constant over time because of cyclical adjustments in the financial system and legislative framework.

In addition, once the initial idea has been exploited and studied, it will pave the way for other companies to start producing similar software without having to spend time on R&D or requirements analysis. If they can make the product more quickly, perhaps streamlining it and maintaining only the basic features, they will be able to compete for the same market at a better price. Once enough companies enter this market, generating products that can be interchanged with one another ("commoditisation"), we reach a unique situation: in the absence of other differentiating factors, consumers will buy the cheaper product, which will generate a highly competitive situation.

This phenomenon is common to any type of product and should also be possible with software. However, certain factors protect the dominant companies in this process, which would ideally lead to greater technological diffusion and bring benefits to users (albeit making it more difficult for companies to obtain large profit margins). As explained above, there are some strong economies of scale on the demand side so it will not be as easy for users to consider competing products as true replacements. Moreover, the use of proprietary formats creates an important captive situation that is difficult to escape from.

Hence, free software emerges as a driving force for a situation in which **free goods are perfectly interchangeable**: the appearance of a similar product that is distributed freely or even free of charge makes it more difficult to maintain high revenues from licensing and may be one of the few ways to break the captive inertia generated by proprietary software.

Free software as disruptive technology

The term *disruptive technology*, coined in 1999 by Clayton M. Christensen, refers to innovations that, for their low price and features or due to their focus on a new type of customer, manage to displace the previous market solutions. Free software could thus constitute a disruptive technology, given the possibility of obtaining it for free and its ability to contribute to the widespread use of software through existing technology gaps.

Though it would considerably limit the possibility of maintaining high profits from licensing, the transformation of the software industry into a scenario of interchangeable goods (commoditisation) could open up new markets, generating an ecosystem of needs around the new interchangeable and widely adopted product.

2) Dependence on economic cycles

Traditional software companies with their product focus can generate huge profits but also suffer major losses during unfavourable economic cycles. Despite being consolidated businesses with established products, between 2000 to 2002, many software companies lost 80 to 90% of their value; even Microsoft lost two thirds of its value (Michael Cusumano, *The Business of Software*).

During adverse economic periods, consumption falls and software products are the first to feel the effects. Users simply stop buying software, which can have a serious effect on the product companies that depend entirely on this source of income. Consequently, it is difficult to find a product company solely of this nature, as the guarantee of its income would be too precarious and unpredictable, and would inevitably suffer in harsh times.

Although any business activity will be affected in such scenarios, companies focusing on services are more capable of maintaining their income due to their long-term contracts and clients – who are mainly other companies and, albeit to a lesser extent, will still need to maintain their infrastructure. In many cases, these infrastructures allow the client company to operate more efficiently, thus increasing its chances of survival in difficult times. As a result, it continues to spend on new technology services.

1.3. Hybrid models

In actual fact, there are many hybrid models that combine the sale of standard products and the provision of services to varying degrees in an attempt to reconcile the two trends. We can consider that the degree to which a company leans towards products or services is indicative of its own life cycle, and there is a widespread trend of a transition to services.

Example of a hybrid company

Consider a company that starts with a pure product model, obtaining high sales and large profits, but which discovers that it is going to be difficult to maintain this level of income. To ensure its continuity or in response to difficult economic times, it may begin to arrange service contracts with some of its customers, witnessing a significant slowdown in the company's rate of growth but obtaining greater long-term stability. The company may eventually put its entire emphasis on to services, having already saturated the market of its original product.

Of course, this is merely a theoretical example, and many companies will not complete or even begin this cycle at the same point.

Furthermore, the transition to services is not an easy one and can have negative consequences if not done carefully. Adopting a hybrid model in response to a crisis, without carefully considering the business strategy, can cause many problems for a product company.

In times of lack of revenue, the company may cede to pressure from different customers to develop highly specific product adaptations that are difficult to integrate with the main standard product. If this practice becomes widespread and the company intends to maintain its revenue through the sale of the standard product, it may encounter difficulties in maintaining compatibility between the new versions released and the specific adaptations for different customers. The work of debugging and development will multiply and can sometimes generate more expenses than revenue for the business.

1.4. Software as a service

The concept of "software as a service" (SaaS) originated in 1999 as a new way to implement software with an emphasis on functionality.

> The basic approach of this idea is that software is important to users insofar as it allows them to solve a problem, i.e. to the extent that it provides them with a service.

Under this paradigm, the need to acquire a software product, have a related hardware and software infrastructure and the installation and support that this requires would be little more than a hindrance to the end user, who has to put up with them in order to obtain the desired functionality.

Under a software as a service model, all of these problems disappear and the software changes from being a product that can be acquired to becoming a service that can be provided. In this sense, it is important to distinguish between the service companies that we described earlier, which **provide software services** (installation, maintenance, etc.), and this new model, which **provides software as a service** (provision of the specific functionality of this software).

To implement this concept, the provider would take care of all the necessary infrastructure, hosting the required software and offering the service on-line through a browser. A sufficiently powerful communications infrastructure is required but the other technological requirements on the receiver side of the service are reduced, allowing the attention to be focused entirely on the functionality offered.

Providing software as a service

More and more companies are using this model to provide enterprise software, such as 37signals with Basecamp (project management tool), and the popular Salesforce.com (CRM or customer relationship management), which allows the software to be tailored to customer needs.

The software as a service model is a low-cost way of providing software to companies, in comparison with the traditional method of selling products. On the one hand, customers save considerable sums on IT infrastructure maintenance and, on the other, providers can offer lower prices because they combine the recurring revenue obtained from the provision of a service and use a single instance of their application at any one time to service a large number of customers.

The presence of both free software and SaaS offers is threatening traditional software vendors, who are feeling the pressure with the entry of these new competitors and will have difficulty maintaining the prices of their products.

Software as a service providers also stand to gain a great deal from the use of free software. On the one hand, using it in their software infrastructure will save them significant sums in licensing or development and, on the other, some companies are using free, GPL-licensed applications to develop their critical business applications, keeping their modifications closed as a way to protect their business differentiation. In this case, they are exploiting a loophole in the GPL: modifications of the code must only be redistributed if the program is redistributed. In the case of software as a service, only the functionality – not the code – is redistributed, so the company has no obligation to share its improvements.

On-line software

We can also find several examples of web applications aimed at private consumers, although this trend is referred to as "Web 2.0". Many have had great success, such as the numerous Google applications and e-bay.

2. Dominant companies in the sector

As we have seen, orienting a business towards products or services will generate very different business dynamics although both approaches can generate profitable business models. Nonetheless, it will be very difficult to keep pure-product companies alive and the barriers to entry will be substantial.

The "Software 500" survey of "Software Magazine" (www.softwaremag.com), which produces an annual ranking of the top 500 commercial software companies by revenue, shows that both types of company discussed here can be found among the most profitable companies.

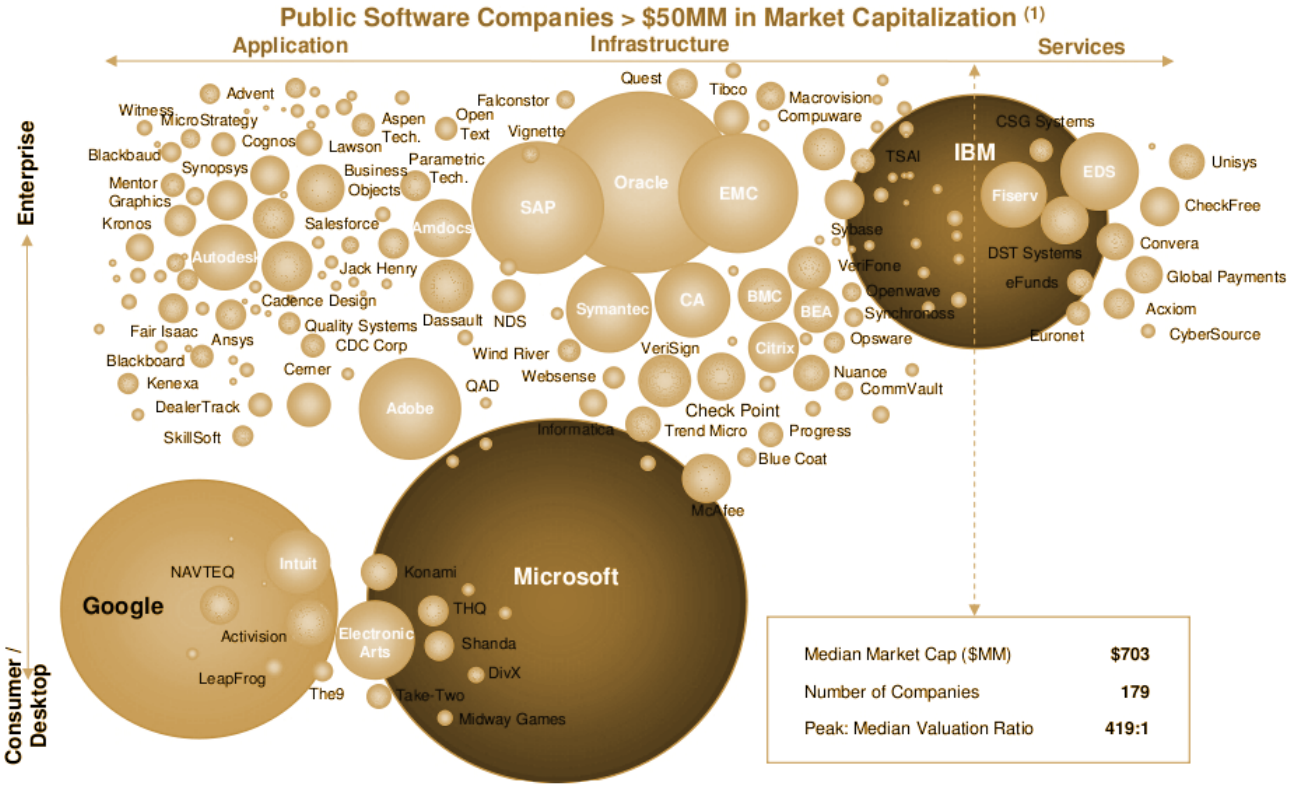
However, of the top twenty, only four have a marked product focus with services representing less than 30% of their total business: Microsoft Corporation, Oracle, SAP and Symantec, which offer leading products in their sectors to corporate customers and mass markets (desktop operating systems, databases, ERP and security, respectively).

Two companies, Lockheed Martin Corporation and EMC Corporation, have a 50% balance between products and services. Of the remaining companies, ten state that their primary business sector is integration, consulting, and outsourcing services, while the rest, although dedicated to the development of specific products, derive their income primarily from the provision of related services.

	Company	Website	Revenue from software /services ($ million)	Revenue growth (%)	Services as a %	No. employees	Software sector
1	IBM	www.ibm.com	$66,451.00	3.0%	72.6%	394,540	Middleware/Application server/Web server
2	Microsoft Corporation	www.microsoft.com	$39,317.00	9.0%	NA	71,000	Operating systems
3	EDS	www.eds.com	$21,268.00	8.0%	100%	118,500	Outsourcing services
4	Hewlett-Packard Company	www.hp.com	$16,918.00	2.0%	92.3%	156,000	Systems integration services/ IT consulting
5	Accenture	www.accenture.com	$16,646.40	7.0%	100.%	140,000	Systems integration services/ IT consulting
6	Computer Sciences Corporation	www.csc.com	$14,615.60	4.0%	NA	79,000	Systems integration services/ IT consulting
7	Oracle Corporation	www.oracle.com	$14,380.00	22.0%	19.7%	56,133	Databases
8	SAP	www.sap.com	$12,309.70	23.0%	29.2%	39,355	ERP (Enterprise Resource Planning)
9	Cap Gemini	www.capgemini.com	$10,158.60	23.0%	NA	67,889	Systems integration services/ IT consulting
10	Hitachi	www.hitachi.com	$9,019.20	5.0%	85.4%	356,000	Storage management
11	Lockheed Martin Corporation	www.lockheedmartin.com	$8,992.00	10.0%	51.2%	140,000	Industrial vertical applications
12	Science Applications International Corporation (SAIC)	www.saic.com	$7,775.00	8.0%	NA	43,600	Systems integration services/ IT consulting
13	NTT Data Corporation	www.nttdata.co.jp	$6,685.80	4.0%	7.9%	21,308	Systems integration services/IT consulting
14	EMC Corporation	www.emc.com	$6,014.50	16.0%	51.2%	31,100	Middleware/Application server/Web server
15	Affiliated Computer Services, Inc.	www.acs-inc.com	$5,353.70	23.0%	NA	58,000	Outsourcing services
16	LogicaCMG plc	www.logicacmg.com	$5,221.40	65.0%	NA	40,483	Systems integration services/ IT consulting
17	Unisys Corporation	www.unisys.com	$4,917.20	3.0%	NA	31,500	Systems integration services/ IT consulting
18	Sun Microsystems, Inc.	www.sun.com	$4,697.00	19.0%	100.%	34,400	Middleware/Application Server/Web Server
19	SunGard Data Systems, Inc. Pvt	www.sungard.com	$4,212.00	8.0%	91.9%	16,600	Financial applications
20	Symantec Corporation	www.symantec.com	$4,143.40	60.0%	3.3%	17,396	Security tools/ Systems

Top 20 companies in the software industry and their main business (prepared using the 2007 "Software 500" study. http://www.softwaremag.com/SW500/

The following figure illustrates the current positioning of these and other software companies by approach (application, infrastructure, services) and the type of customers they target (business or domestic consumers).



Positioning of the leading software companies (with market capitalisation in excess of $50 million and listed on the stock exchange). John Prendergast (2008). "Can Xensource, MySQL or Jboss tell you anything about your company's prospects?". *Open Source Business Conference*. Available at: *http://http://akamai.infoworld.com/event/osbc/08/docs/CEO-CMO-Prendergast.pdf*)

3. Marketing in business: who to sell to?

Thus far, we have looked at several aspects of the basic nature of software companies and the definition of their core activities. However, another fundamental aspect that any company needs to ask itself is what to sell and who to sell to.

3.1. Niche and mass markets

For any company with strong economies of scale, as is true of software product companies, the bigger the user base, the higher the profit margin. Therefore, the seemingly more lucrative option would be to aim its products at mass markets.

However, a strategy like this can be fraught with difficulties: the mass market will be more closely analysed, controlled and saturated by the big corporations. For a company that is just starting out, it will be extremely difficult to compete with companies that are already established and dominant in the sector, and which will also have a large capacity for marketing and diffusion.

It will be easier to meet the needs detected in **niche markets**, which are unattractive to large companies due to their size. For large companies, the potential returns from these markets are too low given the small number of customers, but they will be more than sufficient for a small business. The number of potential niches is vast and there are numerous factors on whose basis we can segment and identify a market. The key question here will be how many potential consumers will this niche provide, as this will allow us to calculate the volume of business and hence, the volume of expenses that the company can afford.

Software offers more interesting possibilities than other tangible products in niche markets because of the absence of geographical barriers with the Internet. A niche detected in a given geographical area may be relatively easily extrapolated to other areas with similar needs or even be extended by itself, without the need for special efforts from the marketing company.

When we create products for niche markets, it is essential to know this particular environment very thoroughly. Besides technical skills, we need to have an excellent knowledge of the activities, priorities and modus operandi of the niche in question. Following Eric Raymond's rule, "Every good work of software starts by scratching a developer's personal itch", it is useful to start with a niche that we form part of, in order to better understand what needs and problems lie within it.

Knowledge of the environment

This is the case of the software developers' niche: it is a well-covered and exploited ground, since every programmer is both a creator and user with an intimate knowledge of the needs and problems of the sector.

Another important factor to consider is whether the product is going to be sold to corporate environments, small businesses or individuals.

Service companies should focus on corporate environments, governments and other organisations because private consumers rarely pay for software-based services. Product companies, however, may choose the prospective clients of the target market based on the features of their products and their business strategy. Corporate customers may be more attractive because they are more willing to pay for a software product and will also contribute to the generation of revenue through services.

In the corporate environment, companies will pay for a software product, but they will also pay for support, training, installation and integration of the product into their existing systems. Companies that purchase software generally pay 15% to 25% of the price of the licence in annual maintenance fees (Dan Woods, Gautam Guliani, "Open source for the enterprise"). They also often seek custom developments to tailor the product to their specific needs. Thus, corporate customers will help software companies to generate revenue from services, giving them more guarantees of continuity. However, these new revenues will be more labour-intensive and the company will require careful management to ensure that the costs of providing the service do not exceed the income generated through it.

Recommended reading

D. Woods; G. Guliani (2005). *Open source for the enterprise: managing risks, reaping rewards*. O'Reilly Media, Inc.

Moreover, support services are often offered for specific product versions, so maintaining services relationship can also help with the generation of revenue in the form of licences for successive versions: although clients have no interest in purchasing the new version, they will be obliged to do so because support for the older version is no longer provided.

The downside is that major corporate clients will be reluctant to hire the services of a small, new company. One of the key factors in hiring is the reputation and trust generated by the company providing the services, so smaller firms or those that have just started up will find it easier to obtain clients of a similar profile, i.e. small and medium-sized companies.

3.2. Patterns of technology adoption and the "chasm"

Detecting a market niche and creating a good product that meets the needs of the group of potential users is not enough to obtain acceptance. To introduce a new product or service, it is essential to take into account the patterns of technology adoption in a group of individuals.

Marketing books traditionally outline a model of adoption based on a Gaussian curve with four groups of users:

- **Innovators and early adopters:** these like technology and innovation. They often adopt a certain product simply because it is new.

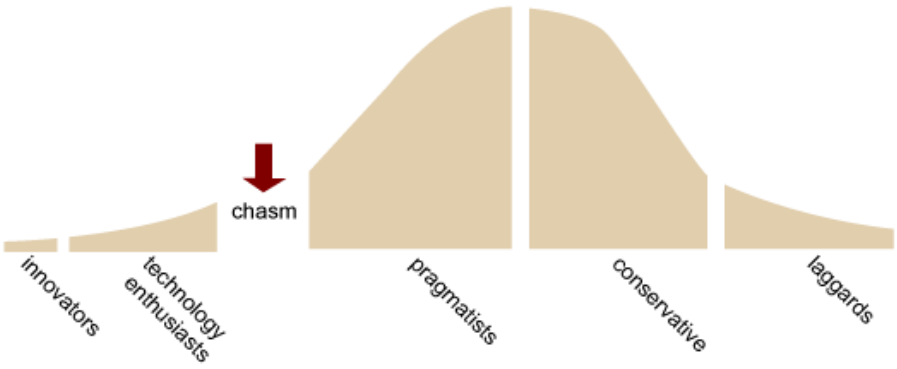
- **Early majority:** these adopt a technology only if it helps them to solve a particular problem.

- **Late majority:** these try to avoid new technologies.

- **Laggards:** these are the last to try something new or may never get to try it.

The curve represents two key ideas: the two intermediate categories cover the vast majority of **potential customers**, and we can only attract the groups in order from left to right (early adopters will adopt it if the innovators have already done so, the early majorities if the innovators have, the late majorities if the early majorities have, and the laggards if the late majorities have).

Geoffrey Moore in his *Crossing the Chasm* renames these groups, calling them **technology enthusiasts**, **visionaries**, **pragmatists**, **conservatives** and **skeptics**, and argues that the theory is flawed because the transition between the enthusiasts and pragmatic majorities is not continuous and difficult to achieve. The early majorities will not adopt solutions that have not been extensively tested but they will adopt those that obtain good references from other pragmatists, so reaching them may sometimes seem like an impossible task. For Moore, there is a chasm between the two groups, so he redraws the curve as shown:



Curve of technology adoption according to Moore

Innovators and technology enthusiasts have a **high tolerance for risk** and the flaws of the new technology because they already have significant technical skills. These users will adopt a technology on the basis of the pure functionality they reveal when seeking innovation. The early and late majorities (pragmatists and conservatives) have a **low risk tolerance** and will be interested in purchasing a product if it increases their productivity but only if it is highly stable and mature.

Thus, an innovative product can be a major success among innovators and technology enthusiasts, but if the creator wants to expand its customer base, it will need to launch a separate marketing campaign, focusing not on the specific features and enhancements of the product, but on generating confidence in it, describing success stories and previous implementations, and indicating numbers of users.

Gaining our first customers in the group of pragmatists and keeping them happy is essential but very difficult, given the vicious circle created: none will adopt a solution not previously tried by other pragmatists.

Confidence can be built by offering integral solutions, which include maintenance, support and training, to attract customers that are sensitive to the stability and user-friendliness of the product. The first customers in this group must be treated with care, with no time or money spared, as they will be the benchmark for the rest. Once we have gained a few benchmark pragmatists, attracting the rest will be a much easier task, and once the pragmatists have adopted the solution, the conservatives will follow without the need for great marketing efforts.

Concentrating on innovators and enthusiasts – on the assumption that, despite being a small potential market, it will be sufficient for a small business – can be dangerous because this group is inherently unstable and will abandon a product as soon as it ceases to be new.

This adoption curve will also mark the life cycle of the product, together with its dynamics of development and marketing practices. The marketing company needs to be clear on the stage it is at and who its customers are at that time, since each group is attracted by very different factors. While adding many new features and maintaining an evolving product will attract innovators, conservatives need the product simply to work in specific scenarios and for it to always do so in the same way. Every change will be a hurdle that they will only be prepared to face if it solves a problem they have.

4. Function of the product: what to sell?

Careful consideration of the type of product to develop is very important. One of the questions we need to ask is whether the product is intended to be an industry leader, follower or a complementary product.

Although being the industry leader may seem more attractive at first, it may not be the most effective approach. When it detects a lack of functionality in a product with widespread adoption, a company has two options: develop its own version with the missing functionality and try to compete with the leader, or build an add-on to complement the possibilities of the leader.

The first option will prove very complex and can easily fail, as it requires a substantial investment not only in the new development but also in the marketing campaign and subsequent sales. In the second, besides the possibility of developing the product in less time, much of the marketing will have already been done by the leader, so it will be much easier to secure adoption of the add-on. Moreover, conservative users (the majority) will be much more willing to incorporate an add-on to a known and proven solution than to change technology and supplier. A common danger is that the leader may decide to incorporate the developed functionality into its core product, thus eliminating the need to purchase the add-on. In this respect, the relationship with the developer of the core product will be essential.

Consequently, it is important to define the role played by other companies active in the sector: which will be direct competitors, which will be partners and which, although in the same sector, will not compete with our product because they have a specific specialisation. By segmenting niches and offering differentiation, we can avoid direct competition from strong companies, and the existence of companies that produce related products or services may be an important factor in our success.

When positioning a product, it is also important to consider the platform that it is being developed for, i.e. which basic set of software will be required to run the product. Consider, for example, the choice of operating system and related technology with which the application will run. This decision will affect the definition of the niche market to be exploited and the type of customer it could be aimed at, but it will also be important for defining our relationship with allies and competitors.

An application designed to run on a particular platform will be a complementary application for that platform. If it is a software package already established on the market and widely accepted, we will also expand the potential market of our customers but reduce the chances of finding allies among the develop-

ers of the platform. The value of these platforms will be largely determined by the number and diversity of applications that can be run on it, so a company trying to establish itself as a platform leader will be very interested in the development of related applications and will hence be a more willing ally.

However, although it is more difficult, it may be better for the company to position itself as leader of a given sector. The question in this case will be whether to try and create a new product category for an untapped niche or whether to try and push out an existing product.

Segmentation and potential customers

For a modest company, the only possibility might be to segment the market until it finds a particular niche in which to position itself. It may be difficult to position oneself as leader in enterprise resource planning applications (ERP), but it could prove easier to develop an ERP for SMEs or for hotel and catering SMEs. Naturally, as we segment further, the competition will decrease, but so too will our potential customer base.

Topping a given market will undoubtedly generate advantages when it comes to positioning oneself as leader and defining the standards that this technology will be based on, but it offers no guarantees. The industry leader is not always the first company to develop a given technology. Sometimes, arriving first and attracting technology enthusiasts can give a false impression of success, since the product must reach the majorities before a company can become the leader. Subsequent strategic and technological decisions will be critical in determining whether the company can capitalise on economies of scale on the demand side to position its product in the number one slot of its sector.

Breaking on to a market that already has a leader will take sales and marketing campaigns that are often outside the scope of recently formed companies. However, the use of a free software product, which competes with a price of zero, can be a sufficiently powerful disruptive agent. In future modules, we will see this and other strategies available to free software for competing on different markets.

Summary

Software needs generate numerous business opportunities throughout the life cycle of the software, from development per se to related services such as installation, migration and user training.

Corporate positioning is key to identifying business opportunities:

- A service orientation provides a more stable economic framework over time.

- An orientation towards product development creates a product economy that is more difficult to maintain over longer periods.

- Hybrid models attempt to guarantee a balance between the above two models.

- The emergence of software as a service is a threat to more traditional models because it offers a more versatile variation for potential customers.

In addition, the exploitation of market segments that are close and familiar can help the business strategy of a new business and with the adaptation of the product to the patterns of technology adoption of the target market.

Lastly, it is also necessary to clearly establish the relationship between the business and its competitors and between its product and that of its competitors. These relationships may even encourage the introduction of the product on to the target market.

Bibliography

Christensen, C. M. (1997). *The innovator's dilemma*. Harvard University Press <http://books.google.es/books?id=SIexi_qgq2gC> [Consulted in April 2009]

Christensen, C. M.; Raynor, M. E. (2003). *The innovator's solution*. Harvard University Press. <http://books.google.es/books?id=ZUsn9uIgkAUC> [Consulted in April 2009]

Cusumano, M. (2004). *The Business of Software* Free Press. Cambridge: Cambridge University Press. <http://books.google.com/books?id=7KAWToDnBAC&dq=the+business+of+software&hl=es> [Consulted in February 2009]

Daffara, C. (March 2006). "Sustainability of FLOSS-based economic models". *II Open Source World Conference*. Málaga. <http://www.cospa-project.org/Assets/resources/daffara-OSWC2.pdf> [Consulted in April 2009]

McKenna, R.; Moore, G. (2006). *Crossing the chasm* Capstone. <http://books.google.com/books?id=GTwFAQAACAAJ&dq=crossing+the+chasm&hl=es> [Consulted in February 2009]

Sink, E. (2006). *Eric Sink on the Business of Software* Apress. New Jersey: Princeton University Press <http://books.google.com/books?id=h5IQuengOGIC&dq=eric+sink+business+of+software> [Consulted in February 2009]