
OPEN SOURCE: STATE OF THE ART

May 2008

Context

Open Source software is leading to a breaking point on the IT market and gives rise to various interpretations, sometimes partisan and contradictory, depending on the players consulted. Pierre Audoin Consultants (PAC) owes its strength to its experience on this market and offers to shed light, as impartially as possible, on open source, in order to see past the myths and better understand the realities of this approach.

Methodology

PAC is the leading European market research and strategic consulting firm in IT, with more than 80 consultants across the World.

For several years, PAC had been conducting a technological and market analysis of open source software, using its experience on the IT market, its databases, its various reports, and its human relations. PAC has met with the various players (both providers and users) present on the open source market and PAC has published reports on open source for the past five years.

This approach has allowed PAC to:

- Define the needs, challenges, and issues related to open source
- Evaluate the offers and strategies of IT services companies and software suppliers
- Understand the market's future developments.

SUMMARY

<u>1</u>	<u>OPEN SOURCE CHARACTERISTICS</u>	<u>4</u>
1.1.	DEFINITION	4
1.2.	TWO KINDS OF OPEN SOURCE PROJECTS	6
1.3.	CHALLENGES & ISSUES	7
<u>2</u>	<u>THE MYTHS OF OPEN SOURCE</u>	<u>9</u>
2.1.	IDEOLOGY AND SEMANTICS	9
2.2.	SECURITY, RELIABILITY, AND DURABILITY	9
2.3.	LICENSES	11
2.4.	MATURATION OF THE MODEL	12
2.5.	COSTS	13
<u>3</u>	<u>REALITIES OF THE MARKET</u>	<u>15</u>
3.1.	REVOLUTION, OR... EVOLUTION?	15
3.2.	STANDARDIZATION AND TECHNOLOGICAL NEUTRALITY	16
3.3.	"GOOD ENOUGH"	17
3.4.	THE SAAS EFFECT	17
3.5.	COLLABORATION	18
3.6.	SHIFTING ROLES IN THE INDUSTRY	19
3.7.	CHANGE IN SOFTWARE DESIGN DRIVES INNOVATION	21
3.8.	OPEN SOURCE STAR-UPS	22
3.9.	PROJECT MANAGEMENT AND IT GOVERNANCE	23
3.10.	BLENDED SOURCE: THE MODEL OF THE FUTURE	24
<u>4</u>	<u>CONCLUSION</u>	<u>26</u>
<u>5</u>	<u>AUTHOR</u>	<u>27</u>

1 OPEN SOURCE CHARACTERISTICS

1.1. Definition

Open source software began as a variant of specific and collaborative development projects, using the Internet to create more or less organized communities. The goal of these communities is to develop software as public property. They are very much involved in research and have contributed significantly to major advances such as Unix, Internet, and, more recently, Linux. It is very difficult to define open source software, as it is a very diverse domain where semantics is often skewed by marketing's effective misuse of terminology.

Definition of Open Source

The expression "Open Source" refers to the freedom of users to run, copy, distribute, study, modify, and improve the software. To be more precise, it refers to four kinds of liberties for user of the software:

- ▶ Run the program, for all uses (liberty 0).
- ▶ Study the way the program functions and adapt it to your needs (liberty 1).

This requires access to the source code.

- ▶ Redistribute copies, thereby helping your neighbor (liberty 2).
- ▶ Improve the program and publish your improvements so that the entire community can benefit from them (liberty 3). This requires access to the source code.

A program is open source if the users have all of these liberties.

Open Source is ruled by a system of licenses that determines how open source code can be used, modified and distributed.

Therefore:

- The code belongs to no one; it is open.
- Open source is not the equivalent of free.
- Community development is central to the model.

This model, in its raw form, is not easily compatible with enterprise IT or with market laws. Few companies have entirely embraced open source or scrupulously respected its principles. The model has therefore been adapted to bring value to companies, but has thus become less “open” insofar as it has come closer to resembling “closed” source software.

Instead of jumping into a debate over “open” and “closed” source software, it makes more sense to discuss a new phase of history in software and IT services. This new phase is the equivalent of the industrial revolution of the software edition:

- The downward trend in prices and a market increasingly centered on volume,
- The integration of production methods from other industries,
- The amplification of information system openness to non-proprietary standards,
- The rise in collaborative software development and the globalization of the market,
- A competition revival on markets that had been “oligopolizing” around a few players and proprietary standards.

1. 2. Two Kinds of Open Source Projects

Projects with a heavy open source component are divided into two rather distinct categories:

- Highly specific projects
- Generic projects.

Highly specific projects are those with a high added value and very specific, where the software brings about strong differentiation. Instead of starting from scratch, the open source makes it possible to embed numerous basic programs that can be completely adapted to business needs, especially for the most critical and performance-demanding environments. It lowers the costs and permit very strong customization. In fact, for quite some time, embedded IT has called upon open source to build its solutions: the French Navy's sonar is based on Red Hat Linux, and one of the media systems on the A380 Airbus is based on Novell's Suse Linux. Open source also allows for these projects to be released from the stranglehold of software supplier updates, or even from the disappearance of these software suppliers altogether. This is particularly interesting for embedded systems in the nuclear arena, where maintenance commitments last 50 years!

The other part is better covered by the media and better known and it concerns projects that call for generalist software. These projects can be just as eminently critical – e.g., a web application server for an e-business during the winter holidays – but they depend much more on standardized programs. It is a highly industrialized, standardized and low-margin market, with software that tends to attack the market via the costs. Most of the open source projects we place in this category center around software such as Linux, Open Office, PHP and Apache. Open source is making the concerned technologies significantly more accessible, but it is also does an excellent job of spurring competition.

1. 3. Challenges & Issues

For companies, every new technology, be it Linux or Eclipse or any new approach – such as open source software – generates challenges and new issues.

Open Source – an Investment Like Any Other

The challenges are easy to spell out – they are the same as those for any technological investment:

- Why choose open source software?
- What are its uses and technologies?
- What are its costs?
- What are its risks?
- What will its impact be on my information system and my company?

And finally, the most important question: what competitive and business advantages will my company gain from open source?

Open Source Software – Investment Remains Specific

The main issue, naturally, involves expected return on investment (both financial and non-financial). Due to the specificity of its model, open source software requires a different approach. In order to come to a decision, it is necessary to weigh the pros and cons of open source software.

The commonly cited advantages of an open source solution are the following:

- Independence from software suppliers,
- The freedom resulting from open source: an adaptability sensitive to the company's needs, software flexibility, independent maintenance, etc.,
- The benefits gained from community development: shared investments, R&D outsourcing, the creation of common interest communities, etc.,
- Significantly reduced – sometimes non-existent – cost in licensing.

The most commonly cited disadvantages to an open source solution are:

- Generally less sophisticated and functional products,
- Generally higher costs surrounding services (training, expertise, set-up, implementation),
- Ambiguous behavior on the part of certain software suppliers, whose “open” approaches are motivated more by marketing than by clients’ actual needs, which results in corruption of the model,
- Difficulty in evaluating the solution’s final cost and surge in complexity

It is more difficult to evaluate other aspects, such as product durability, reliability, the time factor, the inherent entropy surrounding specific developments, security, and support capacity. They most often depend on the user’s project management abilities.

In order to effectively evaluate the challenges and issues involved with open source software, it is helpful first to demystify the concept of open source, to understand its limits in order to better use it. We will see that open source is part of a global technological evolution, and also that it is an unrivaled incentive for competition on the IT market.

We will list the main myths and realities of open source software, confront them with the facts, and evaluate them with regard to the market and to the lessons learned from technological waves that have regularly swayed the IT market in the past.

2 THE MYTHS OF OPEN SOURCE

We believe it is important to demystify certain assertions about open source software in order to better understand what it can bring to a company.

2.1. Ideology and Semantics

The libertarian ideology and the so-called positive values conveyed by open source are very alluring to the public and media. Open source is attractive because it allows for the construction of open systems, but it definitely doesn't have all the advantages that its ideologists attribute to it. The media hoopla surrounding this phenomenon is exaggerated with regard to the realities in the background.

Thus, certain commercial firms claiming to offer open source are misusing the term by calling the entirety of their solutions open source when, in fact, some of their software modules are proprietary, namely to make up for the deficits that can come from open source community developments. And naturally, these modules aren't free... These modules and numerous specific developments made in open source solutions can also lead to a loss of independence with regard to service providers, unless effective project management is put in place.

Choosing open source software solely because it is open source can only be justified in certain rather limited cases. A software choice, especially over a critical platform, should not be motivated by ideology, but rather by realities and nothing else.

2.2. Security, Reliability, and Durability

At the technical level, there are several factors to analyze.

It can be argued that community development is more reliable. Community-developed software necessarily offers more extensive development capacities than could be expected from a traditional software supplier; it could allow for more intensive program testing and also for the faster fixing of flaws and bugs. However, the development of open source software could also be less organized, less often successfully completed, and less "businessy" than developments made by a "classic" software supplier. Suppliers of "commercial open source software" such as Red Hat, Talend, Ingres or

EnterpriseDB partially resolve this issue. This is also the case of the “traditional” software companies that embrace Open Source such as IBM, Sun or Novell. Developers also attempt to bring together these approaches by creating software factories, such as Bull’s NovaForge. End users like the Defense contractor Thales can also create their own open source frameworks as their Theresis.

As far as security is concerned, the source code being shared and exchanged on the Internet is not necessarily proof of security. In fact, if the most critical systems incorporate “open” modules, they are based upon specific developments that are entirely proprietary. This is what has long prevented banks from adopting open source software on their front offices and retail activities.

With the advent of commercial open source, tensions over the open source are increasing. The open source could face the same fate as Unix, where software editors corrupted the project. The so-called ” forks” where open source projects are divided are increasing and they could prevent risk adverse CIO to chose open source.

Rearguard battles among standardization organizations, professional federations, and the Open Source associations do a great deal of harm to open source as a whole, and in particular to its adoption by companies. The tragicomedy surrounding GPL 3 on the basis of anti-Microsoft sentiment and neo-Trotskyism has the worst effect on an IT department that has to make strategic decisions affecting the performance of its company. Likewise, the competition and lack of cooperation among the various software giants have long prevented OpenOffice from breaking into the market. Now that they are cooperating, Open Office is a more viable alternative to Microsoft Office.

The durability of open source software is no more guaranteed than that of commercial software: the community aspect can be reassuring, thanks to continued contact with users of the solution. But if people lose interest in a product, there will be fewer and fewer participants in the community, as is the case for “commercial” software. By definition, commercial software benefits from the financial incentive that makes its customer base valuable and therefore more attractive.

By the way, customer bases never die!

2.3. Licenses

The open source system is built upon not only the concept of openness and sharing of code, but also on the notion of free licenses.

Companies with a software demand need a contract to be sure that their product will work and if not that the problems will be fixed. Community-done software maintenance on projects usually doesn't suffice for company use. In response to this need, a whole range of offers have appeared, dealing with the maintenance of open source-born software and in particular focusing on open source companies built around communities such as Fedora or Apache. The model that works is a model that centers around subscriptions and resembles the classic model of proprietary software suppliers.

Certain software licenses, and in particular the GPL – the best known among them – pose a problem for companies who wish to use them. This licensing system (especially the GPL 3) is extremely virus-prone, which hinders companies who aim to build a differentiating system upon open source software. More doesn't mean better. That also one of the reasons we are seeing the return of FreeBSD, with more convenient licenses for businesses as it has proved in the network of the telecom operators.

Open source can also be a source of problems in legal terms, especially regarding the sharing of responsibilities or issues involving intellectual property, and all the more because the licensing system remains very complicated and chaotic. This greatly harms the success of open source software.

New software are created to help companies to solve this issue such as Black Duck Software, who analyze your code and point out the legal problems you could have.

2. 4. Maturation of the Model

Open source software is a relatively new, and therefore fragile, phenomenon. It is necessary to pay attention and make sure that the contributors' enthusiasm doesn't run out of steam, and that everyone gets something out of it. Therefore it is important to see that the systems should be a real win-win exchange platform. It is not worthy that you give intellectual property and you gain nothing. This is a growing problem. IT companies solve it by contributing in majority to the community attached to their products, and they can also maintain good control on the direction the community takes.

The software heavyweights wish to impose their vision and block their clients with abnormal extensions, often masked by savvy marketing politics. This has already happened on other markets (Unix, Java, etc.). It is a path also used by young offshoots, open source specialists who saw within the model a powerful way of breaking onto the market – by using their financial might to access a significant reservoir of software R&D. This is how Citrix with Xen Source or Sun with MySQL have gotten their hands on the commercial parts of open source projects.

Certain software suppliers, such as Red Hat, Mandriva or Alfresco, are bordering on the open source model, whereas others have left it and since claimed it, as in the case of Open Trust. But it seems that their model is the only way to make money, and they are the models that favor the clients as they give them more security and less support to do internally. Those companies are now established software editors that compete with the other “traditional” software editors. These ones are also more and more “open sourced” as they integrate open source projects in their portfolio, a strategy that was initiated by IBM (Websphere was built like this) and spearheaded by Novell.

In taking an open source approach, it is important to stay as close as possible to the projects themselves and to the standards (which, by definition, can't be bought). If the aim is to adopt commercial versions of open source projects – often more professional – then it is necessary to remain aware that one is dealing with a software supplier, and that it is the company that must profit, even if communications suggest the opposite.

2. 5. Costs

Open source software allows the company to access software developments almost for free, and therefore to outsource part of its software R&D. However, open source programs are not miracle programs capable of covering all IT needs at lower costs, even if this remains one of their main advantages. It all depends on the kind of need at hand.

Open source programs have a different cost structure from that of classic programs. They are programs whose development method may initially present advantages, but whose competitive advantage may ultimately be limited in comparison to more “traditional” software, or even inferior regarding certain needs, especially for highly strategic and complex needs. On the other hand, for other needs, open source is a real alternative. The time factor must also be taken into account: packaged software is often more rapidly implemented and used in the private sector. Time is money especially now where competition is moving with increasing velocity. This is called “IKEA” syndrome: it’s cheaper, but requires more service.

Open source is therefore no cure-all in terms of costs: for example, it can be more reasonable to choose a limited version of BEA Weblogic over Red Hat Jboss, or it can make more sense to go to Suse Linux instead of HP UX.

With this perspective in mind, choosing to have your software maintenance done by a developer rather than a software supplier is not necessarily the best decision. The highly mutualized maintenance offers from “traditional” software suppliers can tip the balance in favor of proprietary solutions. Open source software has the advantage of a relatively weak and very competitive acquisition cost, but simulations of return on investment demonstrate that the total cost of possession after a few years is not necessarily in its favor. This aspect has been key in the surge of open source specialist such as Red Hat that tries to get the best from both worlds: open source with professional support.

Likewise, it is important to keep in mind that open source often tips the balance in favor of systems with a strong specific development component, with the advantages and disadvantages that this involves. Thus, the evolution of specific solutions developed

internally or by a developer can be more problematic than a software product-gearred project.

Choosing open source can be helpful in getting a project approved by your general and/or financial management because it decreases fixed costs (licenses); however, variable costs (maintenance) risk causing damage to the company in the long term. In this regard, it is regrettable that information system directors have less and less of a long-term vision, in large part as a result of the increasing precariousness of their position.

It should not be forgotten, either, that the higher you leap into the strata of an IT system and the further you go in the direction of specific, complex, and critical systems, the greater the services component will be. The price of this services component does not decrease with open source; in fact, it is the opposite.

On the cost side, very packaged products such as SaaS application could also be better priced than Open Source. They also have very standardized and industrialized support and maintenance activities.

As always, there are exceptions, for example for certain administrations:

- Internal IT expertise is not necessarily considered a cost
- The sharing between companies of certain application developments allows costs to be reduced, especially given that these are not activities subjected to a competitive environment.

All IT choices must be made with an all-inclusive approach, where return on investment is calculated on the basis of all the factors:

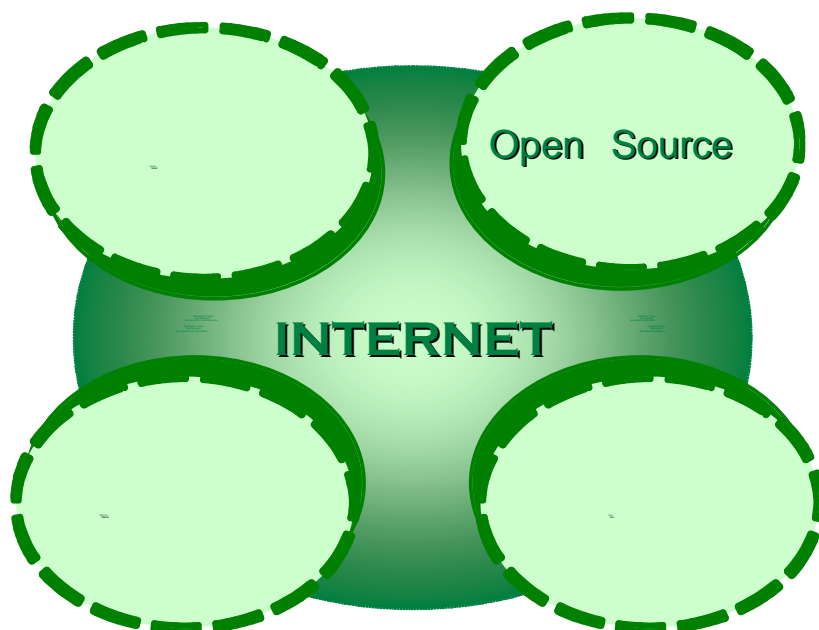
- The lifespan of a program,
- The business improvements it brings,
- Factors that influence costs.

After having demystified of open source software, we will now see that open source has in fact made a significant positive contribution to companies.

3 REALITIES OF THE MARKET

3.1. Revolution, or... evolution?

Open source software is a technological breakthrough much like those that preceded it: it changes certain models, forces players to adapt, and stimulates the market. However, it doesn't revolutionize the market, as did the Server/Client and PC wave that forced certain players such as IBM, Software AG, and Fujitsu to re-evaluate themselves.



Open source must be included in an acceleration phase of the maturity and industrialization (and therefore of price reduction) of the IT market, which manifested itself in three phenomena that affect the three primary aspects of IT:

- Telecommunications: the Internet, reducing costs and making communication networks accessible; the Internet is a catalyst for other phenomena,
- Services: offshore/outsourcing, whether to an external provider and/or abroad,
- Software: open source software, SaaS, and Web 2.0.

These three phenomena were accelerated by the massive adoption of the Internet, which accentuated their effect on the market, especially regarding price and collaboration. So, Open Source, SaaS and Web 2.0. also share a lot in common in the way they are developed and put on the market

Open source software shakes numerous segments of the IT market by becoming a credible alternative.

3. 2. Standardization and Technological Neutrality

Open source software is a powerful advocate of standardization because of its openness. Standards are critical in lowering prices and opening IT systems. Standards are the enemy of monopolies. For any software investment, “open source” or otherwise, the best way is the utmost respect for market standards.

Open source software, well-chosen and well-used, allows for the avoidance of technological constraints related to the offers of certain traditional software suppliers, always tempted by “lock-in” and license systems that can hold back the company. However, this kind of neutrality can also be reached by “commercial” software as long as they respect the official and market standards. SOA approaches are good examples of it. If the source code is open, maintenance is set free from the software supplier’s stranglehold and can be more easily entrusted to third party maintenance. This is great for IT departments, who can then better work the competition to their advantage and get better quality for lower cost.

This is how Eurocontrol, which manages the European air traffic, often uses open source software so that its primary technology providers can interoperate in order that they avoided technology lock-in a highly strategic infrastructure. It also allows them to get the best from the competition from critical custom oriented projects. Another main advantage is independence with regard to the hardware platform (this advantage has eroded with the arrival of virtualization).

On the other hand, open source software can give rise to very strong constraints in the company, which can then fall prisoner to its own choices; as is the case for all software

needs, specific developments around one platform are risky. When this cannot be justified, companies must avoid returning to non-standardized and too customized systems.

Technological neutrality is a key factor in an IT investment. One of the main dangers of open source software is to favor approaches that are too specific; conversely, one of its main advantages is technological independence.

3. 3. “Good Enough”

The notion of the phrase “good enough” is one of the key factors in explaining the success of open source software.

Open source has revitalized basic offers – it is no longer necessary to select a product that is too expensive due its excessive richness, when your need is simple: instead, there is open source software, as well as commercial solutions that have fallen into line with open source prices. It’s the “good enough” effect: you buy only what you need, and you don’t pay for superfluous features. You don’t need a Ferrari to go buy a loaf of bread!

Open source software has redefined and cleaned up the notions of urgency and commodities in IT: faced with open source programs, “traditional” software suppliers lower the price of their products and reinvigorate their offer, by innovating and offering solutions with higher added value, better reliability, in short: more competitive products. Open source software is also a formidable laboratory for testing IT concepts without having to handle significant financial expenses.

Open source has gone to offer solutions in segments of the market where the competition has dried up. It’s the ideal threat to lower software suppliers’ prices, and it’s also a perfectly credible offer in certain segments, e.g. for OS, content management, Internet servers, and office software.

3. 4. The SaaS effect

SaaS, a new breed of ASP (Application Services Provider) has been as Open Source, propelled by the advent of the Internet age. The SaaS concept is competing on the

“good enough” aka banal part of the market with Open Source alternatives. SaaS as OSS open breaches in mature and oligopolistic markets. On the application side and on the SMB market Open Source adoption is impaired by its complexity when faced with SaaS (Software as a Services) products that also surf on the “good enough” concept. Those products enjoy a tremendous success due to their simplicity, liability and cost efficient structure. They are mostly very “closed” source with Sugar CRM as a rare exception and an interesting combination of SaaS and Open Source.

The competition is strong between the two approaches in the office software products: Mozilla and Gmail are the two biggest competitors of Microsoft Exchange, with very similar uses and features. OpenOffice, GoogleApps and some products from star-ups are competing nearly head to head against once again Microsoft's Office. Open Office is mimicking Microsoft products while Googles Apps is clearly positioned on the Social Web/Web 2.0. collaboration tools.

3. 5. Collaboration

Open source software was born, in large part, in university research, and therefore the open source movement is inherently collaborative. The take-off of open source coincided with the spread of the Internet as a primary collaborative tool. Collaboration remains one of the main strengths of open source, as it allows for the quick development, sharing, and testing of programs.

The arrival of open source, a highly collaborative concept, had several effects on the market:

- Need for standardization, in order to more easily exchange and thereby collaborate,
- Revival of collaborative software approaches.

These factors served as a catalyst for the advent of Web 2.0.

The collaboration promoted by open source reinforced the importance of community for software products, whether open source or not. Since then, most software suppliers have been increasingly investing in the enlargement and stimulation of their communities. This is a very good thing, as it helps to better take into account user demands.

The opening of code facilitates the rapid formation of interest communities that allow the initiator (and others) to greatly accelerate the development of software. That same need, developed entirely on a specific and proprietary base, would generally take much longer. Contrary to what we asserted before (the “IKEA” syndrome), in cases where open source software is integrated into a predominantly specific project, it’s a great advantage for the delivery of the project – you’re starting with a preexisting base, so you gain money and time, and then you can unite a community around your project.

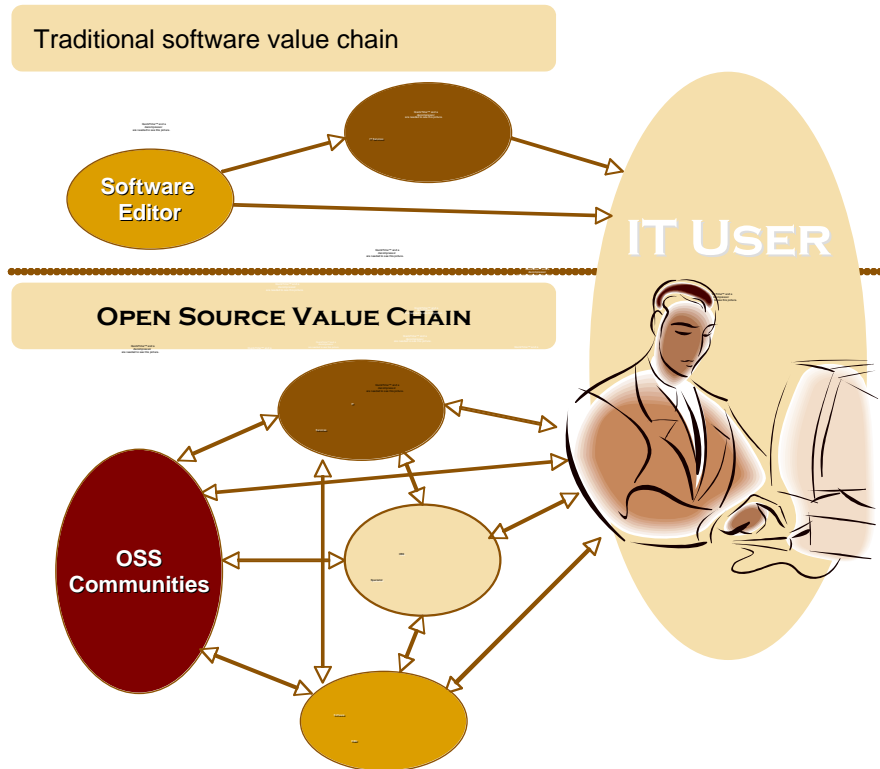
Community size becomes a major issue since it has a direct influence on the collaborative, developmental, and corrective capacities of the software involved. However, this aspect has had a negative effect, especially on market that is often presented as “ethical”: a huge incentive for the market champion. Indeed, the greatest profit lies in the largest communities, those where users go spontaneously, which often reduces competition. This is relatively insignificant in the case of communities surrounding open source projects, but it is unfavorable for communities centered around the commercial distribution of open source projects, as this creates monopoly situations. But as the code is open, competition, even for maintenance, can surge more easily than with “closed source” software. That is the case for Oracle’s Unbreakable Linux, which is based on Fedora, the community of Red Hat.

Collaboration is also encouraged by the technological neutrality that accompanies open source technologies: two competitors can work on the same software base without one of them having to depend on the good will of the other. Thus Thales can collaborate on a CORBA bus with one of its main competitors, Northrop Grumman.

Innovation in IT systems has been strongly stimulated by this collaborative aspect of open source, since it is essential in our network economies to be able to agglomerate various innovations in order to deliver a higher added value solution.

3. 6. Shifting Roles in the Industry

Open source software has allowed for the emergence of both a new value chain and new enterprise models in the professional arena.



The value chain for open source software is more complex, which might be a source of difficulties if your selection process is insufficiently managed. The contractual relationship between the final client and the various stakeholders can become problematic for the different players involved.

Open source software also changes the relationship between software suppliers and service companies. In order to overcome the weakness in sales revenue from open source software, suppliers specializing in open source go for volume and offer services. This is the primary means for “commercial” software suppliers specializing in open source to ensure their endurance in the long term. This model is the most widely spread for OSS and it resembles “low cost” airlines: low-margin but high in volume for industrialized and standardized services. This is very interesting for major systems integrators.

In order to address certain functional shortcomings in products as well as the absence of strong professional grade maintenance capacities for open source communities, IT services companies and open source commercial companies are taking charge and thus coming closer to the software supplier business. Some open source service companies are increasingly placing themselves between the software supplier model

and that of service providers in order to pick up some of the recurrent revenue, and IT services companies are also developing services that increasingly resemble those of software suppliers. They even take on contractual obligations.

In this way, IT services companies are managing to take back the added value that software suppliers had confiscated from them during the surge of software packages in the 90s. This is how IT services companies such as Capgemini, Logica or Bull are forging alliances with companies such as Spike Source, Alfresco or Talend. They can also get involved as a software supplier in the maintenance of open source-born solutions, based on piles of open source or proprietary software.

Many software suppliers, open source and non-open source alike, are betting increasingly on services surrounding their software as a primary revenue model. Service companies, strong in their business knowledge, seem increasingly to be incorporating software that belongs to them in the solutions they offer. These programs can, in fact, be built with the open source software modules and/or specific business components that these IT services companies have developed.

Overall, this increase in added value for services in the IT market has been a strong trend since the beginning of the industry in the 60s. It will accelerate as business needs become more complex. The arrival of services oriented architectures (SOAs), based in large part on component-oriented software development, has only reinforced this trend; IT services companies build programs and software components around their internal framework, and then integrate them into the solutions they offer to their clients. As has been the case in the past, IT services companies are again becoming software solution providers.

3. 7. Change in Software Design Drives Innovation

Through its collaborative and community-oriented nature, open source software has made software designers leave their ivory tower and has brought them closer to market expectations by reinvigorating specific development. It provided them with a whole arsenal of technologies, highly customizable and adaptable to user needs. It has

become one of the driving forces for innovation in those areas where a difference really needs to be made.

Software suppliers increasingly use open source modules in the design of their software. It's an approach that was used by IBM to create its WebSphere offer, with its well-known success. Today, many companies have abandoned the "vertical" design of a product (where everything is controlled from A to Z) and instead adopted an approach based on standardized and simplified modules, often born of open source, where the focus is on differentiating aspects. Rather than develop an application server, McKesson bases its products on Red Hat's JBoss, while IBM uses the Open Office office suite in its collaborative offer, Lotus.

Another of open source's great advancements: "traditional" software suppliers are putting an open source license on their aging/off-market products, or on some parts of their mainstream products to drive adoption. This is increasing the number of available components on the market in order to create specific programs from standardized, free modules. These open source systems also rely on an ecosystem of communities and their commercial expansions. This is how CA separated itself from its Ingres database activity, which became an open source program. Every year, IBM continues to open the source code of a significant number of programs. This kind of change toward open source software allows the community to start up quickly on a sizeable user base.

Young open source offshoots have been able to benefit from the opportunity provided by open source software to break onto a highly consolidated market. Most of these companies have innovated via standardized software offers on very competitive markets. They have also achieved innovation by using all the facets of the Internet and of open source software and by innovating in the way they produce and sell software: buzz marketing, lowered operational costs, outsourcing some of R&D, SaaS, etc.

3. 8. Open Source Star-Ups

But these young offshoots often have problems monetizing the downloads of their free solutions: user reluctance, too weak a difference between the free version and the one you pay for, competition between services companies, etc. With the exception of Red Hat, or some specialists such as Open Trust (which doesn't claim anymore to be "open

source”) or MySQL (acquired by Sun), most open source software suppliers suffer from these problems.

A significant ecosystem of young offshoots specialized in services has also developed, but most of these companies are called to remain very small businesses.

Whereas Europe is far and away the top contributing zone to “open source”, few software suppliers have broken onto the market because the obstructions to creating software-supplying companies remain. One atypical model, such as Open Trust (a services company that is becoming a software supplier) nonetheless works well, thanks to consuming fewer investments, and it is in the process of becoming a huge hit with the changes in the models of some IT Services companies currently underway, such as Linagora in France

3. 9. Project Management and IT Governance

The changes induced by OSS require you to have more governance in your software delivery processes. This fact is reinforced by the inherent complexity of OSS, the relations with communities (a new activity for a lot of companies), the legal challenges induced by the OSS licensing...

Indeed, there are often more specific developments on open source software than on proprietary software. Thus, you run the risk of going back to overly specific systems in which the knowledge of those systems depends on IT services companies and/or individuals within the company, with all the dangers that this involves, especially if you lack a very structured project management.

These specific developments are increasingly industrialized with tools (project management, governance) and methodologies (CMII, ITIL, Six Sigma, etc.). Therefore, it is increasingly possible to make systems cheaply by relying on open source technologies and existing developments. Of course, this must not be a step backwards to entirely customized, hard-to-manage systems, when the trend in IT Systems is going rather towards more buy and less build. Project management is crucial part of the ROI of any IT project, but even more so for an open source project.

The company must assess whether or not a specific development will create more value than a software product. This will depend on the company's economic sector, on its existing IT, on the role of IT in the creation of its added value, on its internal IT capacity, and on its expertise in project management and IT governance.

3. 10. Blended Source: the Model of the Future

The blended source model is the current model of the industry, where closed and open source are mixed: applications build on Web Logic that use Sun's MySQL database or development done with Eclipse and integrated in a Lotus environment. This refers to industrialized specific development based on standardized components. This model resembles that of Dell in industry: specific development cheaper than standardized, relying largely on normalized, standardized, and often-reusable basic components.

This model gets along perfectly with another big concept: Services Orientation or SOA. Applied to software, this model allows for developments that are less costly and more adapted than a standardized solution. This model advocates reusability and customization. Thus, the suppliers of integrated management software products now seek to open their products and to "un-integrate" them by letting them rest on integration platforms – software suites on which the company's applications rest in n-tier architectures.

It makes more sense to view the emerging model as a hybrid of specific development and software products between "Build" and "Buy". It's a model that resembles a Lego set, where you assemble existing bricks – sometimes reusable, and sometimes very specific – in order to create a coherent system. In fact, this is the model used in numerous industries, such as the automobile or aeronautics industries where the "integrators" design the solution, build some of the components and assemble them with components created by other companies.

Open source software is very much a driving force in this development, providing modules and components without requiring payment for licenses. Both open source and non-open source software will be increasingly integrated in mixed architectures. In fact, this is already the case for most open source projects, since the main OS that runs Open Office is Windows, and the main database used by Jboss is Oracle. Thus, the two worlds (open source and proprietary) will mix more and more, even though

some of the beauty of the original concept of open source will be lost. Companies have everything to gain. This “blended source” model will bring the most value in the future.

Just as solid buildings must be built on good foundations, it is well worth it to evolve toward a solid SOA in order to get maximum return on investment from open source components in the architecture. Open source software has to be considered in an overall urbanization approach, where it can be integrated according to its capacities within heterogeneous programs.

It is a model where IT is efficient, adapted, agile, and highly innovative – and therefore, it creates value and can make a difference with competition.

4 CONCLUSION

There is no technological tidal wave in IT. Like any new technological breakthrough, open source software is preceded by a great deal of media hoopla, and generates many expectations – along with many potential disappointments. However, with the promises of standardization, openness, accrued competition, lowered prices, an innovation race, a better correlation between supply and demand, etc., any IT player ought to recognize the benefits of open source for the entire industry. Experience demonstrates that if open source software is a credible alternative, we cannot assert that it constitutes a better choice in terms of quality, security, and costs.

As for any software, open source is a reasonable choice if well reasoned.

Open source software is easier to integrate if you have:

- Extensive experience with specific development,
- Good-sized and/or specialized internal teams,
- Relatively simple needs in software infrastructure, or very specific needs.

To make the best choice, you need to:

- Maintain scrupulous respect for the standards,
- Bet on the side of SOAs,
- Have experience in project management

And, above all, don't reinvent the wheel:

- It often makes more sense to buy an existing program than to redevelop one, even if you use open source software modules extensively.
- Conversely, it can be counter-productive to develop a given program if it exists in the same form in open source.

In order to be strategic, efficient, and reliable, open source software must be understood in the context of a global, IT architecture approach within the company; otherwise, it will be yet another return of IT entropy in organizations.

5 AUTHOR

Mathieu Poujol

Director, Technologies

m.poujol@pac-online.com

QuickTime™ et un
décodeur pour TIFF (non compressé)
sont requis pour visionner cette image.



About Pierre Audoin Consultants (PAC): PAC is the leading European market research and strategic consulting firm for the **Software and IT Services Industry (SITSI)**. We advise IT companies on developing successful end-to-end growth strategies in Europe and in the U.S. through planning, development, implementation, and ongoing support. This is done through market analyses as well as consulting engagements.

Headquartered in Paris, we have been supporting over 300 clients worldwide for more than 30 years. We manage our clients through a specialized local network with offices in Munich, New York, San Francisco, London, and Bucharest, as well as partners in Italy, Tunisia, Latin America (Argentina, Brazil and Mexico), and Asia (Australia, China, Japan, India, Pakistan and Singapore).

For more information please visit our website at <http://www.pac-online.com>.

Pierre Audoin Consultants (PAC)

23, rue de Cronstadt

75015 Paris

Tel: +33 (0)1 56 56 63 33

Fax: +33 (0)1 48 28 41 06

Web: <http://www.pac-online.fr>