How to Eat an Elephant

Arjen Lentz

MySQL AB

<arjen@mysql.com>

1. Introduction

This paper explores a few thoughts and observations that I see as relevant for the marketing of open source in the real world. While many people will know that I work for MySQL AB as a technical writer and trainer, this presentation is not about MySQL. My background includes software development and running my own company, as well as an interest in psychology: I like to observe people and their behaviour, both as individuals and in groups.

Naturally, an observation is subjective, and my thoughts simply reflect my opinion at this time. I have been known to change my mind! I welcome your feedback and encourage additional discussion, that is in fact the purpose.

What you are reading now is not the talk. You will find some of the dry stuff: additional information, background, and references. I also aim to publish and maintain all gathered information and insight on-line.

The talk itself features a wild cast of geeks, customers and our very special guest star, the relevant elephant! If you are a business person, please bring along a developer (aka geek).

2. Analysis

To me, it appears that many proponents of open source, particularly developers, believe that open source does not actually require much in terms of marketing effort or product development. Seen as a form of evangelism, the open source message is regarded to be so compelling that, once delivered, the audience will *"see the light"* and convert. But that is, overall, not how the real world responds – to the great dismay and annoyance of the evangelists!

We know Free and Open Source software is a fast growing market, and many of us are involved as developers, primary vendors, solution or service providers, or consultants. We are confronted with existing terminology, decision processes, analytical methods, all of which are very much part of a "*closed source*" environment. Habits change slowly, so we must try to deal with the current realities of this marketplace while also trying to shape it more to our liking.

We are looking at a long term process, with lots of little cautious steps that are actually big milestones. The size and speed of these steps is dependent on the individual user. Luckily, it is possible to group the users along a number of boundaries, but I think it is important to realise that each of these groups will need to be catered for individually. So we are looking at quite a diverse market.

What product are we trying to sell? By sell, I mean "get others to use" - irrespective of whether money is involved. There is no single product, as not all open source is the same. Open source can be used in a development model, as well as in a business model. There are quite a few possible combinations and I think that they really can not (and therefore should not) be lumped together. In addition, is it prudent to exclude closed source products – polarising the market into open source versus closed source offerings – or could it make more sense for us to not focus specifically on that particular aspect of our offering?

2.1 What Makes a Product

Historically, developers in the open source realm have written software to satisfy their own needs. An excellent example of this is Apache, originally developed by a group of web masters[1]. This is fundamentally different from today's open source situation, where we see many more users of open source products and a much smaller number of developers. I believe this is a sign of maturity, as we have to accept the simple fact that not everybody is or has the desire to be a developer. This does, however, change the way things work. If you exchange code with a peer, documentation and ease-of-use is less important. And let's not forget support. But ordinary users have very different expectations!

Sometimes, concern is expressed that companies make money with someone else's hard work. Such companies sell a product, or a combination of a product and services, based in whole or in part on open source components. Is this concern justified?

I think that a tarball (archive) with sources (even if it compiles and installs cleanly) does not make a product. There are many tarballs out there, but much fewer products. Often, the

tarball merely contains a component (library) that can be used for writing either bigger components or complete packages. But to create an actual product, a library or package must also provide an easy installation process, documentation, support, and a marketing channel so potential customers may learn about its existence and merits. I would also mention distribution, but we know that distribution is a virtual freebie with the Internet.

However, I would suggest that many very useful components are not actually commercially viable as independent products. A company having to acquire the rights to use a number of different components would also find the cumulative costs prohibitive. It would raise the cost of the end product, which inevitably decreases the size of the potential market. In short, I think we can conclude that if these particular components weren't in essence gratis (regardless of them being open source), they – and consequently the derived products – would simply not exist at all. That in itself is an excellent point to think about: open source provides a lot of enabling technology. Most such components would be licensed under LGPL or a BSD-style license.

A company that uses open source technology to develop a complete offering that is commercially viable has, I think, put in quite a bit of effort. I do not have references that quantify the relative weight of the components versus the rest, but I do know that when you have developed a working prototype, you're still very close to the beginning of the product development process. So I would assume that in this case it is actually the company that has put in the vast bulk of the work. Please note that this does not diminish the importance and significance of the component developers in any way.

In fact, it brings up other interesting ideas. Developers of different components can pool together and share the costs of human resources enabling for instance user interface design, documentation, and product marketing. In many cases, that may lower the costs enough to make the individual components viable as (cheap) products, and provide returns to the actual developer. It addresses the different market we see today, as well as the changed role of the developer. However, it is a significant challenge, as developers have strong opinions.

2.2 Licensing

Licensing is another issue that elicits strong emotional responses, there is a significant philosophical difference in between for instance GPL licensing as promoted by the Free Software Foundation, and BSD-style licenses. In some ways, you might say that these differences are as great as the divide with closed source.

I believe there is no single best license, one or more options will be suitable for a particular situation and a choice will have to be made. This evaluation may even include closed source licensing.

If you fully own your code, or only use LGPL and BSD-licensed components, you are able to make your product available under a compatible open source license as well as under a closed license. Second-generation open source companies such as Sleepycat and MySQL have built a viable business model around this so-called "dual licensing" concept. Over 50% over MySQL AB's incoming is derived from its licensing[2], with the remainder made up by supplementary services.

Some products may only be viable in a pure closed source model. Yes, sorry. I'm afraid that no amount of philosophical debating and wishful thinking actually changes this right now. But it is generally believed that the boundary is shifting, and more software becomes suitable for open source licensing. Predominantly, commodity markets show this: first operating systems, then database servers, open source appears to be crawling up the software stack. But considering office suites (word processing, spreadsheets, presentations) and the like, I think that the commodity base is also clearly widening.

Sidenote: it has been suggested that the software market was ripe for commoditisation anyway[3], i.e. it was not triggered by the rise of open source. In that case, it may in fact have been commoditisation that has enabled open source – which appears to favour business models that thrive in a commodity market – to grow. It may not matter, but I found it interesting anyway.

I have also personally come to the conclusion that it may often be more appropriate to sell a specific product, rather than a grand concept such as open source. For example, products such as Red Hat Enterprise Linux are accepted by companies that otherwise have great fear of open source. Rather than pushing the conceptual point, Red Hat focuses on the merits that the product provides. I would recommend the same. If the customer makes a positive choice for this product, the business relationship will allow further discussion over time.

Some of the advantages of open source that often get cited:

- Trust: no code is hidden, and you present an open face.

- Security: more eyeballs may detect possible problems quicker.

There are of course more possible advantages, but it is important to realise that not all are applicable in every product or situation. Carefully consider which will be relevant for your target market. Your particular solution is likely to involve a combination of licensing leverage (code, trademark, etc) and supplementary services. There is a nice page with "101 ways to make money off open source"[4] which may also give you some ideas.

2.3 Orphans

Certain closed source products have reached the end of their commercial life, and the community encourages companies to release the code under an open source license. This has for instance happened with some old operating systems and legacy applications, where there is also an active community.

Other companies however release the source code for some of their existing software when it no longer makes (enough) money. The company may even do a press release claiming that it is now actively engaged in open source. But without significant community interest, I would call this process orphaning and that is not really what open source is about. It may still have some merit, as the code is not completely lost. So a user may find benefit from it, or it may be of interest for research.

2.4 Development

Open source is not simply a license, it is a development model. Existing traditional companies may have to adjust their internal structure significantly to be able to cope with it, and this process may be painful. New start-ups probably have a significant advantage.

From the above, we have already seen that you can't simply tag something as "open source" and be successful. If you are thinking about developing a new product, open source may be an interesting option to consider.

Certain aspects of software development apply regardless of the licensing. One such lesson is that if you present a selection of your target market with a very early sample of your product, and listen very carefully to their feedback, the final product will be of higher quality. This concept is explored in detail in chapter 11 "Product-development practises that work" of the book "Strategies for E-Business Success"[5] that is now a few years old but which I think still offers a lot of valuable insights.

The early sample of the product will have only limited functionality. This is important! I have often heard from a person or a company working on a piece of software, and it is their intention to release it under an open source license when it is ready. In addition to this, no customer gets to see the product before it is finished, it is developed entirely behind closed doors. While it is obviously scary for a developer to show unfinished work to potential customers, I would strongly recommend they overcome this hurdle and take advantage of the opportunity. It is of course vital that it does not turn into an early customisation process, but it is obvious that aspects of the design are more easily adjusted (or even completely changed) when much of the code has not yet been written.

As an example, it was in fact Microsoft that used this strategy when first developing Internet Explorer. Bundling with Windows aside, the product was quickly accepted by users because it was already tuned to their wishes at the time. I won't make any comments on where it went after that – the point is, this one aspect of the development of Internet Explorer can help us make more successful products, and so it would be silly to disregard it.

Combined with the additional benefits of open source, code quality and security for a first commercial release can be significantly improved, and then further developed. The development and licensing model of MySQL, for instance, provides excellent feedback and a large and varied test base, delivering clear advantages both for commercial licensees as well as GPL users[2]. New features are made available early in the development process of a MySQL alpha release.

Please note that the book does not end up recommending to release new versions at high frequency. From its research, it finds that the feedback cycle for different versions may distract developers and complicate the process. The first trial is definitely the most important. I think this is very relevant for open source projects, many of which tend to be "trigger happy" with new product versions. It will, however, also depend on the organisational structure. You have to decide what you (and your users!) can cope with.

2.5 Focus and Diversity

One key element that I think is often missing from open source projects is focus. The developers try to solve lots of things for a wide audience, and end up producing a product that does not really suit anyone or is too complex (remember the magic "bloat" word). The problem is basically that the developers have not

thought at all about who their prospective target audience might be. Once you think about this and have decided, it is almost certain to make a product look very different indeed. Naturally you don't want to specialise too much, by all means maintain flexibility (we love our plug-ins!) but focus is absolutely essential. Define your target market, define the scope of your product, and create a solution for the specific problems.

Originally I had planned to talk about "diversity" under a separate heading, but it actually blends nicely with "focus". Because if you focus, this means you will not target the entire market, and so openings remain for other offerings. Think for instance KDE and GNOME. Perhaps not the best example as I don't think either of them has fully worked out their focus and stuck by it, but they do seem to appeal to different groups (with some overlap). Some call this "conflict", and there appears to be a lot of that in the open source arena. I will continue to call it diversity, as I see it as a good thing. Investigating different solutions is I think an excellent way to innovate. Some ideas will flourish, others will be assimilated into a larger project or disappear completely.

Perhaps it's a tough challenge, but next time someone someone asks you "what is the best Linux distribution?", why not talk about the diversity? Also think about what kind of user they are, and limit the options you present, e.g. Gentoo is probably not the best solution the average home user.

3. Customers vs Users - What Market Share?

The (current) real world often asks "what is your market share?" and traditional estimates of market share are sales-based, a method that is clearly flawed with Open Source products on the scene. In terms of "*installed base*" you may be huge, but chances are that in terms of "*units sold*" you are peanuts, small fries, or just plain non-existent.

What it comes down to is that market research papers will not acknowledge the role you play in the market, simply because your "*units*" are gratis or much cheaper than those of your competitors. And to make things worse, chances are that your competitors are very reluctant to disclose how many units they actually sell. Don't be discouraged! First of all, while continuing to explain that this measurement does not work well for open source, we must cope with the current reality. So how can we do this?

For many products, market share can be proven in a different way and in a some cases this has already become accepted in the market – that is an encouraging sign. One key example is Apache. The proof there is provided by companies like Netcraft, and I don't think the other vendors contest their assessment that Apache powers the vast majority of the world's *active* websites[6].

It may appear to be more difficult to come up with numbers for others products, but there are solutions. MySQL has been using a rough estimate based on the number of web servers that are running PHP (again, using Netcraft statistics). This is not MySQL's only market, and the result (currently an estimated 5 million users) may be widely off, but people reckon it's probably a low estimate. So it is quite acceptable, and it gets quoted with confidence all the time.

MySQL has many more *users* (the 5 million estimate) than *customers* (a known number of 5000). So for this particular product that uses free distribution, the ratio users:customers would be 1000:1. Again, the estimate may be off, but a) it is probably conservative and b) at this scale the numbers remain impressive no matter what.

Don't be afraid of devising your own methodology, as long as you are open and clear about how you came by your numbers.

3.1 MozOO.org

This initiative brings together a number of concepts discussed here, and can be regarded as a kind of "proof of concept" for me.

MozOO.org[7] is a CD with Mozilla and OpenOffice.org as its main components. It also includes Sun's Java Runtime Environment, Macromedia's Flash player plug-in, and dictionaries for both Mozilla and OpenOffice.org. It is for Windows only, and it includes an installer program. The original idea was developed by my wife Harmony (not an IT person) and myself, and then made reality with the assistance from many others. They are of course all credited on the MozOO.org website.

The base product is an ISO image of about 110MB, so small that it will actually fit on a MiniCD! That has turned out to be an unforeseen but very nice marketing advantage. It may also serve as a lesson that if you are developing a CD-based product, perhaps you do not want to fill up the CD!

The reason we decided on the above combination is that it has clear focus: it addresses the needs of a specific target group – I would say home users and small businesses –

with tools they all use or need: an Internet browser, e-mail program, and an office suite. All of the components on the MozOO.org CD are available independently, but it is not easy to get it all installed and working properly. Even the order of installation is significant.

MozOO.org also combines open source products with proprietary ones, the licenses were such that I was able to put all the components onto this CD without conflict. This puts the pragmatic approach to the test: while a pure open source solution may be preferable from a philosophical viewpoint, I looked at the needs of the prospective users and decided that I would not be able to resolve the specific problems without using a combination of open source and proprietary components.

MozOO.org targets a specific market, and we foresee it will have a limited lifetime as this market develops. That is fine with us, as we are trying to solve problems, not keep the product alive at all costs. But what problems and needs does this product actually address? I have already mentioned the ease of installation. Without MozOO.org, the individual components might not be used by many for whom they could be useful. The needs are security, stability, virus vulnerability, spam control, open standards and standards compliance, and last but not least: choice. These are all topical things, providing viable marketing angles for this product.

MozOO.org is not meant to compete with other initiatives such as Knoppix, TheOpenCD and GnuWin. I believe they serve completely different audiences. I hope that MozOO.org will help Mozilla and OpenOffice.org gain more ground on the widely used Windows platform (accepting the fact that many users will not, at this point in time, switch to Linux). I think that MozOO.org is one of those small steps I wrote about earlier. It a significant, but small enough to be considered by someone who probably would not be reached via other means, at this point in time.

Already, a few companies in Australia use MozOO.org to serve their customers. It releases them from the burden of developing a bundle in-house, and it eases the deployment process. That is great. I am very interested in discussing other ways to get this product to the target audience. Essentially consumers, accessing the appropriate distribution channels is not necessarily easy.

3.2 Open Source Industry Australia (OSIA)

Another recent development is the founding of Open Source Industry Australia[8]. Instigated by Con Zymaris who had already successfully started Open Source Victoria (OSV) last year, a meeting of interested parties was held during Linux Conference Australia in Adelaide and we were able to get the initiative started since. OSIA gathers the forces of open source related businesses on the national level, and works together with AUUG, Linux Australia, and other organisations. The focus of OSIA is on business, but also government and education.

But rather than only think big, why not also think really really small? Instead of only looking the big customers, think about the following facts[9]:

- small and micro business (< 20 employees) account for 96% of all businesses and employs almost 50% of the workforce.
- Micro business (< 5 employees) itself accounts for more than 80% of all businesses and employs more than 25% of the workforce.

This market is huge potential, particularly for open source offerings as they are a) fairly low cost and b) generally offered by local small businesses. This is, essentially, how it works already. They reach local businesses and often directly into the home. Things that are accepted, spread rapidly. We just need to find ways to make even better use of this advantage.

It occurred to me that like individual developers, open source businesses can pool their resources – this time for marketing, and operated somewhat in reverse. Naturally, press exposure plays a part in OSIA's agenda, this bundles the resources towards an external target. But marketing is all about getting your key messages across, plugging them whenever possible. If open source businesses together develop a set of key marketing messages, which they then individually use in their own advertising and customer contacts, the results could be very interesting indeed. Similar to a brand name (also franchises), people would see the key messages often, in different places, contexts and presented by different businesses. This way a large number of small businesses could, over time, have a big impact by working together.

Again, it is a long and possibly slow process, but I think there are excellent opportunities that require goodwill more than money. We can retain our goal for world domination and I'm

sure we'll reach it, but it will probably take us beyond next Tuesday.

References

[1] How Apache Came to Be, http://httpd.apache.org/ABOUT_APACHE.html

[2] The MySQL AB development and business models, http://www.mysql.com/company/
MySQL business background (AUUG 2003 conference)
http://dev.mysql.com/Downloads/Presentations/stateofdolphin-backgrounder.pdf
http://dev.mysql.com/Downloads/Presentations/mysql-state-of-dolphin-2003-09-04.pdf

[3] The Industry Standard: Guest Blog: Ross Mayfield, Tech Commodities 101, http://www.thestandard.com/movabletype/rossmayfield/archives/000399.php
Commoditized software: fact of fiction? (ZD Net), http://zdnet.com.com:80/2100-1107_2-5272787.html

[4] 101 ways to make money off open source http://www.freeroller.net/page/ceperez/20030611

[5] Book (2001): "Strategies for E-Business Success", Erik Brynjolfsson & Glen L. Urban, editors, MIT Sloan Management Review, center for eBusiness@MIT (http://ebusiness.mit.edu/), ISBN 0-7879-5848-4

[6] Netcraft (surveys of what software web servers run), http://news.netcraft.com/

[7] The MozOO.org project website, http://mozoo.org/

[8] Open Source Industry Australia (OSIA) Limited, http://www.osia.net.au/

[9] Australian Bureau of Statistics - Year Book Australia 2003, http://www.abs.gov.au/