Diploma Thesis / Diplomarbeit

# Synergies and Opportunities: Open Source and Commercial Vendors

A study of the relational database market.

Version 1.0.1 - November 13, 2006

Lukas Kahwe Smith
Mat. No. 188220
<smith@pooteeweet.org>

Technische Universität Berlin
Informatik und Gesellschaft
Sekretariat FR 5-10
Franklinstraße 28/29
10587 Berlin, Germany

Prof. Dr.iur Bernd Lutterbeck
Prof. Dr. Christian Wey
Matthias Bärwolff

# Revision History

I would like to thank anyone who has provided additional feedback after my initial public release on November 3, 2006.

| Version | Date | Changes |
|---|---|---|
| 1.0.0 | October 30, 2006 | Initial version |
| 1.0.1 | November 13, 2006 | Minor typo fixes based on community feedback |

Table 0.1: Revision History

# Acknowledgments

First, I would like to thank all of the participants of the email interviews (Appendix A) listed in order by their respective first names:

> Gavin Sherry (Alcove Systems Engineering), Holger Klemt (HK-Software), Josh Berkus (Sun), Joshua D. Drake (Command Prompt), Kaj Arnö (MySQL AB), Lenz Grimmer (MySQL AB), Luke Lonergan (Greenplum), Richard Hipp (Hwaci), Rick Hillegas (Sun), Ulf Wendel (MySQL AB).

I would also like to thank the following people who have assisted me in one way or another while writing this paper, again listed by their respective first names:

> Andreas Nitsch, Anika König, Armin Ortmann, Prof. Dr. Christian Wey, Christopher Jones, Dan Scott, David Fetter, Eike Schilling, Georg Richter, Jan Strotmann, Josh Berkus, Jutta Horstmann, Matthias Bärwolff, Meike Rademacher, Ralf Lippold, Rodney Smith, Tobias Erbsland, Torsten Logemann, Zak Greant.

# Executive Summary

This paper analysis the influence of open source in the market for relational databases (RDBMS). It examines primarily the economic but also in parts the legal and social dimensions. In order to do so, it first gives an introduction to the theory of informations goods and open source. Following is an overview of the RDBMS market and the SQL standards. The core of this paper is a series of case studies which illustrate how both individual developers but also companies are influenced in their abilities by open source. The paper also features a number of email interviews with members of several open source database projects. In this process it was discovered that open source can assist developer in their career development. Companies on the other hand profit from lowered transaction and development costs. Open source also assists in building markets for complimentary products or even to undermine core markets of competitors.

Diese Diplomarbeit analysiert den Einfluss von Open Source im Markt für relationale Datenbanken (RDBMS). Dabei wird primär die ökonomische, aber auch in Teilen die juristische und soziale Dimension betrachtet. Hierzu wird zunächst eine Einführung über die Theorie von Informationsgütern und Open Source geboten. Anschließend folgt eine Übersicht über den RDBMS Markt und des SQL Standards. Kern der Arbeit sind eine Reihe von Fallstudien, die aufzeigen, wie sowohl einzelne Entwickler, als auch Firmen durch Open Source in ihren Möglichkeiten beeinflusst werden. Hinzu kommen eine Reihe von Email Interviews mit Mitgliedern diverser Open Source Datenbank Projekte. Dabei wurde festgestellt, dass Open Source für Entwickler durchaus als Karrieresprungbrett dienen kann. Firmen wiederum profitieren von reduzierten Transaktions- und Entwicklungskosten. Open Source kann aber auch dazu dienen Märkte für komplementäre Produkte zu entwickeln, oder gar Kernmärkte von Konkurrenten zuuntergraben.

Diese Diplomarbeit wurde in englischer Sprache verfasst.

# Contents

# List of Figures



# List of Tables

# Chapter 1

# Introduction

A bold statement on market evolution states: "Commoditization is something that happens to every successful industry eventually" (Murdock 2006, 91). Commoditization is the process whereby technology that was previously owned exclusively by a few vendors, and therefore priced at a premium, becomes widely available at comparatively low prices. Due to its open nature, open source software could likely be a critical factor in facilitating this development. This paper aims at examining to what extend this statements holds true within the software market and more specifically the RDBMS software market. But more importantly, the purpose of this paper is to study how individuals and companies can leverage open source in mature markets.

Oracle, IBM DB2, MS SQL Server and to some extend Sybase can be considered to be the "Big Four" proprietary RDBMS of today (Babcock 2005). All these proprietary database vendors have made their solutions freely distributable even for commercial use and open source is said to have been a factor in the appearance of these versions (Berkus 2006). While these are closed source stripped down, some say crippled (Garry 2006), versions of their non-free parents, we are also seeing previously proprietary solutions being open sourced. These RDBMS may not have the same large following of the "Big Four", but these solutions do cover the vast majority of the features of them. The market is also being "eaten" from the bottom up by the ever-growing strength and scope of open source RDBMS (LaMonica 2005). So technology that used to be available only in proprietary solutions can be found in open source software packages which are closing the feature gap quickly (Horstmann 2005).

## 1.1 Open Source

It can hardly be disputed today that open source has had a considerable impact on the software industry. What Richard Stallmann initiated after suffering from missing printer drivers (Williams 2002) as a political movement to ensure that all software was to become "free as in freedom" (Stallman 2005) has become a major paradigm shift (O'Reilly 2004) in large parts of the industry. All leading software companies have acknowledged this in some form or another. IBM has pledged considerable money, patents, software and other resources (IBM 2006) to open source. Novel has restructured itself around open source from the ground up by acquiring two of the major open source software companies Ximian and SUSE (Cowley & Peréz 2003). SAP (SAP 2006) and Computer Associates (Vaas 2004) respectively have both released major software components as open source as well. While Microsoft probably still is the main opponent to open source, they have none the less realized that they need to deal with this new challenge. Microsoft has launched fierce marketing campaigns attacking the open source Linux operating system (Kuchinskas 2004). But they have also created an internal project team to study the open source phenomenon (McConnachie 2006). More importantly, even Microsoft has released code under their own open source license (Microsoft 2005).

On a simple level open source software can be identified by being licensed under one of the many Open Source Initiative (OSI) approved licenses (OSI 2006*a*) that follow the open source definition (OSI 2006*b*). Key aspects of this definition are the right to distribution, the access to the source code and the ability to create and distribute derivative works. However, reducing open source to a novel licensing scheme is a too limited view. There are considerably more dimensions to note here. In practice open source has resulted in collaborative development efforts that span across different companies, organizations, continents and ethnic backgrounds. What we are seeing is that open source is proposing an entirely new way of thinking about intellectual property (IP) rights (Weber 2004). Open source is increasingly mentioned as one of the main driving forces behind software commoditization (Murdock 2006). This obviously has grand economic implications that require entirely new business models to exploit and entirely new strategies to compete against. In a world of radically reduced transaction costs and globalized markets (Shapiro & Varian 1999) open source offers a radically new approach to customer relations and the innovators dilemma (Christensen 2005). We see the lines blurring between vendors and customers as open source provides a social solution to collaborative development (Raymond 2001). This is especially significant considering Lawrence Lessig's observation that "code is law"

(Lessig 2000). It also provides new examples of how companies manage to "cross the chasm" (Moore 2002). Of course open source adoption is not without risk, but there are ways to manage the challenges in adopting (Woods & Guliani 2005) and creating (Fogel 2005) open source. Politics has lately also taken notice of the potential to regain more control over their information technology (IT) infrastructure where they previously had little options but to rely on often foreign closed source software packages (Weber 2004).

Simultaneously, the scope is increasing as the open source idea is spilling over to other areas. For example, open source has already begun to span into the hardware world (Sun 2006). However, it is also making its way out of the IT ecosystem. One of the most prominent examples for this trend is the Wikipedia project. It brings together researchers from all fields in order to create an open encyclopedia that is giving long established incumbents huge headaches to compete with (Waldman 2006). Additionally, the success of the Creative Commons license (creative commons n.d.) is leading some people to ask: "Where (if anywhere) are the Boundaries of the Open Source Concept?" (Updegrove 2006). Universities are also noticing the potential to adopt the open source process in fields outside the computer sciences (Lerner & Tirole 2002). Even the US government is seeing the potential. Recently, it released documents in Arabic to the public hoping that amateurs would assist in the translation (Bray 2006).

Open source covers legal, social, political and economic dimensions and an ever increasing scope. Discussing the entire depth is obviously beyond any single paper. Possible sensible approaches therefore mainly focus on analyzing parts either horizontally or vertically. This means either address a single dimension in any number of markets or alternatively focus on a single market looking at any number of the mentioned dimensions. I have chosen to do the latter. Specifically this paper will be looking at the economic dimension of the relational database software market. As part of this analysis I will touch on the legal, social and to a lesser extent on political dimensions to examine the economic interests of companies as well as individuals in the open source community. This for example means that this paper will not attempt to cover the benefits to society at large in any detail.

## 1.2 Relational Database Market

The relational database market is particularly interesting. For one it is among the oldest software components. The relational database concept was proposed as a way to solve issues with navigational database management systems in 1970 by Edgar Codd while working for IBM. In the RDBMS market we are seeing many different business models from purely proprietary, to hybrid proprietary and open source and finally to pure open source. Some open source database companies have even acquired previously proprietary software to make them open source. We can also see companies with very different sizes competing in this market. On one end we have large general purpose vendors such as Microsoft, IBM and increasingly Oracle. On the other end we have small to mid-sized companies like MySQL AB and EnterpriseDB. Finally, there is even a very small one man company called Hwaci that provides support and add-on services around SQLite.

The RDBMS market does have some limits in its ability to provide a complete example for open source. While the database market is huge (Darrow 2004) it is not an end-user product market. Instead, databases are usually integrated deep inside software products by software developers. From this follows that studying the database market will likely only provide limited indications about end-user interaction with open source. At the same time, it does give an opportunity to study an informed market where customers choose their solution based on a thorough technical understanding. However, as noted before the market is not small which provides the necessary room for a considerable diversity in the market. Another point to consider is that since the database market is so mature, the rate of feature innovation cannot be expected to be very high (Powell & Moris 2002). This does mean that any process that is still able to push innovation in this area is particularly impressive. So even given the fact that databases do not provide a good study on end-user interaction or huge innovation leaps, it does serve as a good example for many challenges in software and knowledge development in general.

## 1.3 Key Questions

If open source is making a significant entry into the database market, a detailed analysis of the effects of open source on the market for relational database software is

needed to better understand the causes and implications. In what way does economic theory validate open source business models? More importantly, what kind of business models are used by open source companies in this market and what challenges do these create? Are there any factors in the RDBMS market which make it easier or harder for open source to compete? Similarly, the question arises what enables the open source databases to compete with the current and former proprietary solutions? Why do all major proprietary vendors offer stripped down free for commercial use versions today? Why did some owners of proprietary solutions decide to forgo their exclusive control over the source of their databases? What was their aim, what did they stand to gain and what did they really lose by this step? More importantly, how successful were they in achieving their goals? Aside from what the business world stands to gain, the question is also what does the open source community stand to gain from working together with proprietary vendors and corporate sponsors? How have and how should vendors of proprietary solutions as well as open source communities react to these emerging challenges based on the previously mentioned points? Finally, to what extent has the database market been commoditized and what role if any does open source have in this process?

## 1.4 Research Approach

In order to answer these questions I will be building up a theoretical basis for discussion. As a primer I will give a short introduction to the economics of information goods. Following will be an overview of the open source phenomenon and a look at the value proposition of open source with an emphasis on the business world. This will include a mainly theoretical look at the relevant legal, business and social aspects. Furthermore I will be looking at the historical developments inside the relational database market and the SQL standard. This will enable a better understanding of the deciding factors which either enable or prevent open source from succeeding in this market. After establishing these fundamentals, I will examine several case studies to illustrate ways in which businesses have interacted with the open source world. The first case studies will focus on the PostgreSQL and SQLite communities and how they have benefited from business involvement. The dual-licensing model will be analyzed using MySQL AB as well as the acquisition of InnoDB and Sleepycat by Oracle as showcases. The proprietary-extension[1] model will be examined by looking at

[1]The term "proprietary-extension" as a business model was created as part of this paper, due to lack of a better available term, to describe companies that extend open source with proprietary code.

the commercial companies working within the PostgreSQL ecosystem. The final set of case studies will deal with the open sourcing of Interbase, Cloudscape and SAP DB respectively. These case studies will highlight the challenges and opportunities in creating and maintaining a community and the question of how open source can be used to attack commercially more successful competitors. All of the case studies will touch upon support, since it's such a universal business models. Within each of the case studies I will examine how the previously introduced theory is manifested and what lessons can be learned as a result. Aside from printed literature and online Internet publications the case studies will draw from email interviews sent to several key figures operating in the open source RDBMS market.

# Chapter 2

# Economics

## 2.1 Introduction

Software falls into the more general category of information goods. In the post industrial information age businesses have increasingly shifted their focus toward these types of goods and more specifically in the last decades into software. This market has grown to a triple digit billion dollar market according to IDC (McPherson 2002). Therefore, it comes as no surprise that economists have paid attention to the implications of information markets. The good news is, even though there has been changes in technology "somehow, the basics laws of economics asserted themselves" (Shapiro & Varian 1999, 1). That being said, it is still important to identify how these "basic laws" relate to software, and open source in particular, in practice. Also, there are some key differentiating aspects to information goods which need to be understood. In this chapter we will explore the information good economics in general and in the following chapter we will look at open source specifics. This chapter is largely based on Shapiro's and Varian's book "Information Rules" (Shapiro & Varian 1999). Numbers in parenthesis behind statements in this chapter reference page numbers in this book.

## 2.2 Information Goods

One of the main aspects is that information comes with "high fixed costs but low marginal costs" (3). So much so that the cost of reproduction is effectively zero[1]. This trend has been pushed even more by cheap large media formats like the CD (20) followed by the proliferation of the Internet (22) pushing down transaction costs to near zero as well (48, 100). However, one thing to remember is that while costs have been dropping "the dominant component of the fixed costs of producing information are sunk costs, costs that are not recoverable if production is halted" (21). Therefore, "the key to reducing average cost in information markets is to increase sales volume" (28). Related to the low cost of reproduction and distribution, combined with the ability to make perfect copies of digital goods, creators of information goods have found it difficult to effectively control their intellectual property rights in practice (4).

Consumers are increasingly suffering from "infolation" (Brauch 2000) simply because they are exposed to so many channels of information. Companies selling information goods therefore face huge challenges in getting their products into the heads of consumers. Another point to note is that information goods are so-called "experience goods" (5). That is, in order to value it one must effectively consume it. This is less the case with software since contrary to a news paper where you do not know in advance the contents of today's edition, in software you are likely to buy multiple copies of the same software product or service.

Software is also "notoriously subject to switching costs and lock-in" (11). Where consumers usually face little challenges by switching from one car brand to another or even owning multiple cars from different makers, the same cannot be said about software. Products that are "subject to strong network effects tend to exhibit long lead times followed by explosive growth" (13). This means that "growth is strategic imperative [..] to achieve the demand side economies of scale generated by network effects" (14). It is therefore often necessary to make large upfront investments in order to "tip" the market in your favor and to benefit from demand side economies (14). While its obviously important to have a strong competitive product, it may be even more critical to control what the public perceives as the likely future market leader

[1] Yaron Ilan disagrees with this oversimplification. He notes that software makers do suffer from "increased marginal cost of distribution" in their quest to reach the minds of their customer base (Ilan 2006). The fact remains however, the cost per unit is much reduced compared to previous conventional distribution and marketing techniques.

(14). However, the reduction in distribution costs makes it somewhat easier to fight these network externalities (189).

## 2.3 Competition

Information, and software markets in particular, tend to have very low barriers to entry. Whereas a car manufacturer requires elaborate physical production facilities, complex logistics and distribution mechanisms, there are no such requirements in the software market. With cheap Internet access it is possible for a group of young software developers to build a serious new entry to the IT market from their homes within a few months such as Alando in Germany (later bought by eBay) or Flickr in the USA (later bought by Yahoo!). Due to rapid advances in technology, as well as the limited patent protection for software, it is additionally difficult for companies to be able to shut out new competitors. That being said a lot of these companies then get scooped up by larger companies. Rasmus Lerdorf, creator of the popular web scripting language PHP and infrastructure architect at Yahoo!, even created a series of talks labeled "Getting Rich with PHP" as a tongue in cheek to this trend (Lerdorf 2006). Cisco's CEO John Chambers even goes so far to say: "We don't do research - we buy research!" (284).

However, there are still ways for market leaders to secure their positions. Due to the fact that information goods have strong network externalities, a "market leader often tends to be cost leader" (30). This means with new products growing the user base is the number one priority rather than aiming at making money (Moore 2002, 192). In order to be able to initially ignore making money, software makers can exploit the cheap marketing and distribution over the Internet. But even that has limits and so they turn toward venture capital:

> "All too frequently, even when they are led by experienced managers, enterprises that are funded for long periods of time fall into a 'welfare state mentality', losing their sense of urgency, and looking for their next paycheck to come from yet another round of financing instead of from the marketplace." (Moore 2002, 197)

Moore also notes that without profitability startups remain at the mercy of these venture capital firms (Moore 2002, 196). Even sufficient funding they still need to realize

that "you simply cannot spend your way into the hearts and minds of technology enthusiasts and visionaries" (Moore 2002, 198).

That being said, once a software maker has established himself in the market with a solid user base he can now exhibit the benefits of demand-side economies of scale. That is, users will be more likely to choose the product not necessarily due to reduced prices as the result of supply-side economies of scale, but instead because so many other users, with whom the potential buyer intends to network, are using the same product. More importantly, "unlike the supply-side economies of scale, demand-side economies of scale don't dissipate when the market gets large enough" (180). Metcalfe's law explains the phenomenon as follows: "If there are n people in a network, and the value of the network to each of them is proportional to the number of other users, then the total value of the network (to all the users) is proportional to n x (n-1) = $n^2$ - n" (184). While the exact formula is questioned frequently (Briscoe, Odlyzko & Tilly 2006) "the main point of Metcalfe's Law is that the value of networks exhibits super-linear growth" (Simeonov 2006). In order to build networks companies need to find their entry in the market by "forming alliances, cultivating partners, and ensuring compatibility (or lack of compatibility!)" (10).

Of course at one point the company also needs to figure out a profitable business model. For software the two main options are selling shrink wrapped products or services. In reality most companies provide a combination of the two. The economist Hal Varian clearly favors selling products over services: "The large fortunes of the information age lie in the hands of companies that have successfully established proprietary architectures that are used by a large installed base of locked-in customers." (135). In regards to service oriented business he notes:

> "This strategy is somewhat dangerous for two reasons. First, if your customers really need technical support, they may decide your product is of low quality. Second, technical support is very costly to provide. Promises to offer support that are not delivered can be disastrous in terms of public relations" (61).

Nonetheless, there are still ways to provide support services at reasonable costs. For example, just as news services might charge extra for fresh information (56), software makers can give additional information and security bulletins to customer willing to pay extra without having to generate this information for every customer individually. Another service frequently offered is consulting and training. The latter is often provided in the form of certifications. If executed properly even Varian agrees "the strategy of selling complementary products or services to your installed base has

the very attractive feature that it can be executed profitably and successfully while enhancing, rather than jeopardizing, the buy relationship, and while encouraging customer entrenchment" (160).

The obvious advantage of providing good services is that it locks your customers into a feature that is otherwise hard to quantify. Aside from obvious comparable facts, like service hours, its very hard to compare two service contracts, while there is a tendency to solely rely on feature lists when choosing an off the shelve software product. Happy service contract customers therefore face huge risks when switching. Where as they can test a competing product, its is close to impossible to reliably test the associated service level they are likely to get once all the contracts are in place. Also, depending on the type of license sold, a customer could end up using a given software product indefinitely without ever upgrading which is not the case with service contracts which need to be renewed (167). Even when selling software with time limited licenses this just means that "the customer is least attached to the incumbent supplier when most of its equipment is nearing the end of its useful lifetime" (119). The key aspect therefore must be to provide a good balance through useful upgrades along with services and training which slowly lock the customer into that product by moving the customer away from standards and toward proprietary technologies.

Useful in this case means that both the features and the price point must match the customers needs. Obviously one of the goals from locking in customers is to become able to keep prices high to ensure high margins while discouraging competitors from entering the market: "This is what economists call limit pricing: set prices as high as you can without encouraging others to invest the sunk costs necessary to enter the market" (30). As every customer has different needs it will become necessary to provide these different customers with a product that provides the features they need or that they are supposed to become dependent on (32). Power users realize they will have to pay more to get all the features they need, while customers that do not require all these features are the first targets for potential new market entrants. Being able to realize this kind of perfect price discrimination is hard to achieve in practice. However, by providing different versions for these customer groups, also known as versioning, companies can maximize their profits while reducing opportunities for their competition. Ironically, stripping down a product in order to provide a cheaper alternative product for normal users often requires additional investments (63). Finding the right combination of features to provide to each of the different customer groups is a huge challenge in and of itself. The Internet has improved the situation here as well (36). Customers can provide feedback with little effort. In some cases the application can even generate and transmit the relevant information auto-

matically. Features such as these often collide with consumer privacy laws though. Maintaining a close relationship through service contracts can also help in determining customer needs.

## 2.4 Coopetition

In a fast moving industry like the software market, innovation and how to provide this to the market is a key business aspect. Questions about how to manage intellectual property rights and standardization can determine success or failure while trying to build up a critical mass behind any given product and underlying technology (16). How closely should one protect innovations from competition, how fiercely should one look for unlicensed use: "The instinct to seek out and charge all those who use copyrighted material runs deep and can easily cause otherwise sensible executives to defend their rights past the point of economic return" (89). At the same time, companies find themselves in the weird situation of having to cooperate and compete with companies in the market at the same time: "The term coopetition captures the tension between cooperation and competition prevalent in network industries" (228). Companies must realize that in order to grow there is an alternative to taking market share from the competition and that is expanding the size of the market itself: "We think that the natural tendency is for producers to worry too much about protecting their intellectual property. The important thing is to maximize the value of your intellectual property, not to protect it for the sake of protection" (97).

For the reasons mentioned in the previous section, customers are weary of lock-in. Even worse investing in a new technology that fails to gain a critical mass can turn a promising innovation into a dead end when the producer drops the product to cut its losses (230). Without support and updates companies who have invested into such technology have no other option but to write off the license fees and training costs. A product that is based on an open standard therefore has a much better chance at attracting customers. As a result, companies can choose to open up their innovations to their competitors to grow the market by building more customer trust. This makes particular sense if no company is strong enough to force the creation of a de facto standard (199). Usually standardization includes giving up some intellectual property rights either through a non discriminatory license agreement among members in a closed standard consortium or the world at large in case of an open standard.

Competing in a standardized market requires building a strong brand name, exploiting manufacturing advantages and providing complementary products that are beyond the scope of the standard (201). Obviously depending on the standard, there is room to innovate in the algorithms that implement the standards to enable companies to build up a competitive edge. For the most part however, "standards shift competition away from features and toward price, for the simple reason that many features are common across all brands [..] For this very reason, consumers tend to seek more extensive standards than do suppliers" (231). At the same time, the "loss of variety", creation of "inferior technology" due to excessive compromises and the lost savings due to lack of necessity for "aggressive penetration pricing during a standards battle" are all disadvantages for consumers as a result of standardization (233). Varian also observes that similar to the classic "tragedy of the commons" standardization can even slow down innovation: "Just as few took the trouble to protect the common grazing land from overuse in the seventeenth century, few today will make major investments to advance a technology that is in the public domain" (255). At the same time, companies who feel strong enough might decide to end the "truce" setup by a given standard by extending, modifying or even attempting to replace an existing standard to once again increase the competitiveness in the market.

## 2.5 Management

Obviously successful management will need to deal with the challenges mentioned in the previous sections. In addition, two books describe common stumbling blocks for technology oriented companies. These are magnified due to the rapid speed of development where market leaders all too frequently have not even existed a few years before. In "Crossing the Chasm" (Moore 2002) Geoffrey A. Moore speaks about how a company needs to transform internally as well as in customer communication in order to make the transition from young innovator to a mature company that is able to reap the profits of it's innovations. Clayton Christensen coined the term "disruptive technology" in his book "The innovators dilemma" (Christensen 2005) where he explains that companies that do not manage the pace of innovation and new emerging technologies and markets eventually outrun themselves.

Moore comes from a marketing perspective: "Within an ever-changing society, marketing represents the ongoing effort to keep the means of production - our products and services - in touch with the evolving social and personal conditions" (Moore 2002,

xiii). Just as Varian, he understands that controlling the market expectations is at least as important as providing a solid product (181). Varian even goes so far to call this a "frightening" proposition to any technology advocate. Moore also notes the challenges created for marketing by the deteriorating relationship with customers due to campaigns that rely on "fear. uncertainty, and doubt" or "FUD" for short (Moore 2002, xvi). Another issue with advertising "is that it is a one-way mechanism of communication. As the emphasis shifts more and more from selling product to creating relationship, the demand for a two-way means of communication increases" (Moore 2002, xvi). Good communication is vital though especially when it comes to addressing the biggest challenge in customer relations:

> "To be specific, the point of greatest peril in the development of a high-tech market lies in making the transition from an early market dominated by a few visionary customers to a mainstream market dominated by a large block of customers who are predominantly pragmatists in orientation. The gap between these two markets, heretofore ignored, is in fact so significant as to warrant being called a chasm, and crossing a chasm must be the primary focus of any long-term high-tech marketing plan." (Moore 2002, 5)

This means that the same innovative people that have created the product initially will become a liability once it is time to serve the broad market as "their fundamental interest is to innovate, not administrate" (Moore 2002, 199). Suddenly "R&D is driven not by the laboratory but by the marketplace" (Moore 2002, 208). As a further example Moore points out that with all the focus on high-tech features the need for skillfully crafted packaging is overlooked (Moore 2002, 209). This relates to another important aspect of software production: "Productization means making software work for the general case and making it as easy as possible to use" (Woods & Guliani 2005, 21). We will look at this topic in more detail in the following chapter.

Clayton on the other hand focuses more on the relation between innovation and competition. He observes that companies that execute perfectly all too often still find themselves outdone by the competition. This is due to the fact that "the pace of technological progress can, and often does, outstrip what markets need" (Christensen 2005, xvii-xviii). This opens up the chance for new products to enter the market which are good enough at what the market really needs, but that bring with them an entirely new set of features that are needed for a market that may not even exist yet (Christensen 2005, xviii-xxiv, 264). This is what he calls a "disruptive technology":

> "First, disruptive products are simpler and cheaper; they generally promise lower margins, not greater profits. Second disruptive technologies typically are first commercialized in emerging or insignificant markets. And third, leading firms' most profitable customers generally don't want, and indeed initially can't use, products based on disruptive technologies." (Christensen 2005, xx)

This is significant because precisely these characteristics open the chances for competition to enter the market:

> "In their efforts to stay ahead by developing competitively superior products, many companies don't realize the speed at which they are moving up-market, over-satisfying the needs of their original customers as they race the competition toward higher-performance, higher-margin markets. In doing so, they create a vacuum at lower price points into which competitors employing disruptive technologies can enter." (Christensen 2005, xxviii)

In this context the term "commoditization", the process of previously exclusively owned technology becoming mainstream, is important to remember. The key lesson to learn from this is that established companies must realize that they "cannot expect [their] customers to lead [them] toward innovations that they do not now need" (Christensen 2005, 258).

# Chapter 3

# Open Source

## 3.1 Introduction

Today open source is a well-studied phenomenon in economic terms. It is now clear that it is not simply an altruistic or even communistic movement and that it can be described in known market terms. However, this realization has still not made it to the world at large. This might be caused by the fact that "to an economist, the behavior of individual programmers and commercial companies engaged in open source projects is initially startling." (Lerner & Tirole 2002) Eric Raymond defines open source as "the process of systematically harnessing open development and decentralized peer review to lower costs and improve software quality." (Raymond 2001, xi). The open source development process and business models based on it have the potential to dramatically effect all aspects of knowledge generation and distribution:

> "I became interested in open source as an emerging technological community that seemed to solve what I see as very tricky but basically familiar governance problems, in a very unfamiliar and intriguing way. [..] By experimenting with fundamental notions of what constitutes property, this community has reframed and recast some of the most basic problems of governance [..] If you believe (as I do) that software constitutes at once some of the core tools and core rules for the future of how human beings work together to create wealth, beauty, new ideas, and solutions to problems, then understanding how open source can change those processes is very important." (Weber 2004, vii)

Therefore studying its implications is critical not only to the IT market, but to the entire information society.

Open source software is produced and distributed in a unique fashion:

> "The hacker culture and its successes pose by example some fundamental questions about human motivation, the organization of work, the future of professionalism, and the shape of the firm - and about how all of these things will change and evolve in the information-rich post-scarcity economies of the 21st century and beyond." (Raymond 2001, xii)

The most obvious difference is the choice of license that instead of focusing on defining very rigid allowed usage patterns highlight the freedom of the licensee and sublicensees. Among the key characteristics in development is often the lack of a single physical location where development takes place. Short release cycles and modular architectures are other notable features. These attributes determine the behavior of open source in relation to commoditization, innovation and productization. This leads to a specific set of possible business models, but more important to a very significant impact on society.

## 3.2 History

In the early days of software development code was shared quite freely, often without an explicit license. In academics it is common practice to share results so that others can reproduce and extend the scope. But even in the commercial realm software was not seen as a source of income (O'Reilly 2004). Software was very deeply linked to specific hardware. Manufacturers even facilitated sharing of software code. Most software also had to be shipped with source anyways. Due to the differences in hardware and the lack of standardization most software had to be compiled for every specific setup. Distribution of software in binary form was therefore often simply not feasible and as a result a natural exchange of source code with modifications emerged. IBM used to call source code additions coming from their customers "field development" (Woods & Guliani 2005, 7).

The most drastic example that illustrates a shift from these early days is the advent of the IBM compatible PC. IBM was a heavy weight in hardware, however they had "missed the boat" to get into the home computer market. In an effort to close the gap they choose to use standard off the shelf components to assemble their computers and let Microsoft, a young startup at the time, deliver the operating system. IBM did not understand the implications of their decisions at the time (O'Reilly 2004). Due

to standardization of the components, there was now an opportunity to make considerable money from software that ran on top of these standardized components. Suddenly the software was the driving piece that could command a premium as the underlying hardware market become commoditized: "Whatever open source developers' feelings about Microsoft, they should acknowledge Microsoft as the natural parent to their own brand of commodification" (Asay 2006*f*, 105)

However, a young developer at the Massachusetts Institute of Technology's Artificial Intelligence Laboratory was unwilling to accept being unable to fix, extend and generally tinker with the software he had to use. A new printer was donated from Xerox to his lab, but it failed to work in the way he wanted and he found Xerox unwilling to provide the source. Richard Stallman attributes this as one of the key moments that drove him to create the GNU software project and its now famous GPL free software license. He defined free software in terms of four levels (counting from zero in true "hacker" fashion):

> "First, Freedom Zero is the freedom to run the program for any purpose, any way you like.
>
> Freedom One is the freedom to help yourself by changing the program to suit your needs.
>
> Freedom Two is the freedom to help your neighbor by distributing copies of the program.
>
> And Freedom Three is the freedom to help build your community by publishing an improved version so others can get the benefit of your work." (Stallman 2001)

As can be gathered from the rhetoric in the just cited speech: This was about more than a pure technical issue. This was about ethical software development. Eric Raymond was one of the key people behind documenting the early free software days and claims to have been one of the people that inspired Netscape to release the source code of their popular Netscape Browser (Raymond 2001, 173). He described the members of this movement as "his tribe". This notion of ethics and tribal community inspired a lot of the negative reactions from the business world. It was considered impossible for such a group of long-bearded ethical nut cases to produce any even remotely complex yet reliable piece of software from their homes. The movement has been equated with a "cancer" (Greene 2001), "IP socialism" (Sanders 2005) pushed by "sort of communists" (Kanellos 2005) with intend to destroy market value of all software. Stallman however coined the term "free software" to describe his movement.

However, it should be noted that "some of the characteristics that governed the early days of open source have changed now that open source has become popular" (Woods & Guliani 2005, 18). Furthermore, "open source code does not obliterate profit, capitalism, or intellectual property rights" (Weber 2004, 3). In order to rid the process itself from this negative connotation and to give the numerous developers a better term to associative with, several key members of the community got together and coined the term "open source software". Eric Raymond even goes so far as to say that "it seemed clear to us in retrospect that the term 'free software' had done our movement tremendous damage over the years" (Raymond 2001, 175). Within less than a year it became the widely accepted term to use. When referring to both free and open source software it is now common to use the acronym FLOSS. For the purposes of this paper we will use the words free software, open source and FLOSS without distinction.

The early days of open source were not without trouble. One of the key developers on the poster child open source project Mozilla that emerged as part of the open sourcing of Netscape quit after a year with the words "open source is not magic pixie dust" (Raymond 2001, 63). Corporate endorsement of open source and Linux in particular expanded nonetheless: "Thus, it was the mid-July announcements by Oracle and Informix that really closed out this vulnerable phase" (Raymond 2001, 183). Internal memos inside Microsoft, which became known as the Halloween Documents (Raymond 2006), showed that open source was now a serious contender even to the core software of the largest software manufacturer in the world. Today "the examples of corporate success for open source would fill a phone book" (Woods & Guliani 2005, 2).

All major software and hardware manufacturers have accepted open source as a major movement in the IT market and have published their own, donated to or assisted open source software in some way. This even includes Microsoft, the main perceived adversary of open source. Tim O'Reilly even goes so far to say that "most of the 'killer apps' of the Internet, applications used by hundreds of millions of people, run on Linux or FreeBSD" (O'Reilly 2004). BIND should also be mentioned here as it is one of the key infrastructure components of the world wide web. This means that essentially everybody that uses the web at least indirectly uses open source. Another distinction that needs to be made clear is that the opposite to open source is not commercial software. Instead, the opposite is what is called proprietary software, which highlights the fact that this software is not distributed in source. This is the key distinction to open source software. It is however possible to sell open source software. Finally, open source software can be run in conjunction with proprietary software. For example, the popular web server Apache runs not only on open source operat-

ing systems such as Linux and FreeBSD, but also on various proprietary variations of UNIX and even Microsoft Windows. That being said software industry heavyweights like Microsoft, Oracle and SAP are still selling their proprietary software very successfully.

## 3.3 Licenses

The legal basis for open source software relies on specific aspects of the licenses employed. As a matter of fact, the "Open Source Initiative", OSI for short, defines what criteria a license needs to fulfill in order to be considered an open source license (OSI 2006*b*). These fundamental principles can be summarized as follows: "Open source licenses must permit non-exclusive commercial exploitation of the licensed work, must make available the work's source code, and must permit the creation of derivative works from the work itself" (Laurent 2004, 8). Especially the anti-discriminatory provisions make it clear that this definition is also politically and morally motivated (Laurent 2004, 9).

From this definition it is clear that open source licenses define an entirely new property regime: "The conventional notion of property is, of course, the right to exclude you from using something that belongs to me. Property in open source is configured fundamentally around the right to distribute, not the right to exclude" (Weber 2004, 1). As a result, the term "copyleft" has become popular to describe the fact that "open source IP focuses on keeping code access open rather than closed" (Asay 2006*f*, 106) in contrast to copyright law that attempts to do the opposite. Power is derived from reputation rather than ownership. However, open source still obviously needs to work within the established laws and this is where open source licenses come into play.

While proprietary software licenses usually tightly control all uses of software as well as prohibiting modification and any redistribution, open source licenses focus almost entirely on the latter: "The requirement to release modifications to a given open source project is triggered by distribution: so long as the code itself is not actually distributed (and only the resultant service dictated by the code is), open sourcing of modifications is not required, but voluntary." (Asay 2006*f*, 91) As a result, some of the largest users of open source software like Google and Amazon find themselves in quite a comfortable position. They are not required to make their proprietary software, or even changes to the open source software they use, open under an open source license. This is due to the fact that they do not distribute software, but instead

provide services on top of proprietary and open source software. However, that does not mean that these companies do not choose to do so nonetheless in some cases. This kind of business model is known as "application service providing" or ASP. We will revisit this topic in a later section of this chapter. However, it cannot be stressed enough that open source licenses make no restrictions on using and even modifying software for internal use. It is even allowed, Linux distributions are a good example of this, to sell open source software. Any restrictions are solely triggered by distributing modified versions.

It should be clear that there are a large number of licenses that have been approved of meeting the OSI definition. While these licenses obviously share a number of similarities there are key differences that result in very different real world usage scenarios. Even if this is not part of the OSI definition, most contain warranty disclaimers (Laurent 2004, 13). That being said, court cases about defects in proprietary software are fairly seldom as well (ifrOSS 2005, 21). Open source licenses can be categorized into three groups: strong propagation (for example the GPL), weakly propagation (for example the LGPL) as well as non-propagation (for example the BSD license) (SSC 2006)[1].

The term "propagation" refers to the type of restrictions placed on the distribution of modified versions of the software. Strongly propagating licenses require that all derivative works, including those that merely incorporate the given software, also be licensed as a whole under the same or a similar open source license. Weakly propagating licenses are less restrictive only requiring that modifications to the software itself have to be licensed as open source. Non-propagating licenses do not make any requirements to the license chosen for any derivative works, but usually require that the original software is credited in the derivative work. As a result, weakly propagating software can be incorporated into proprietary software, while non-propagating software can even be turned into proprietary software. Strongly propagating software however cannot be combined with proprietary software. As a result, these type of licenses have also been called "viral", because just like an infection all software that is combined must also become open source.

It should also be noted that any software can be distributed under any number of licenses. Licensing a software under multiple licenses is know as "cross-licensing" (Laurent 2004, 162). This approach helps alleviate issues where different open source

[1]The cited document was updated within less than 3 months after initial publication. Most notably the term "infectious" was replaced with "propagation" `http://www.groklaw.net/articlebasic.php?story=20060313054547209`.

licenses might be incompatible. It is also used to optionally provide open source software under a proprietary license. The practice of using a strongly propagating license as well as a proprietary license for the same software is known as the "dual-licensing" business model that we will examine later on. It should be clear however that any combined piece of software can only be made available under a specific license if all the creators agree. This makes it increasingly difficult to add another license to an existing open source project with a vibrant developer community.

There is a wide range of established licenses to choose from that cover the full spectrum of what is possible within the open source definition. As a result, "it is almost certainly a mistake to try to draft your own open source license" (Olson 2006, 83). Writing a new license will require the assistance of a lawyer that needs to know how to draft an international applicable, easily understandable license. But even then, it will require a number of successful projects that make use of this license before its characteristics will become known enough among users to be considered trustworthy (Laurent 2004, 176). While buyers of traditional proprietary licenses are faced with numerous different licenses, which need to be understood before acquiring a given piece of software, the challenge is much smaller when investing into open source software employing a common license. Organizations only need to evaluate any given open source license once, this reduced transaction costs significantly, especially for companies that do not have their own legal team to consult.

In order to understand how open source licenses work in practice it is important to realize that the forming of a contract "rests on two fundamental assumptions: one, that there is some mutual obligation created by the agreement, which is known as the consideration, and two, that there is mutual consent, or a meeting of the minds, as to the terms of the contract, usually described as the offer and the acceptance" (Laurent 2004, 148). The obligation is usually met in open source licenses by requiring the licensee to include the license and credits when distributing copies (Laurent 2004, 148). The consent requirement is fulfilled by the fact that open source licenses do not restrict use in any way for licensees. The license is really only concerned about distribution of modifications which requires a license. So unless the person has acquired another license, the only rights to distribution are granted via the open source license. Ergo, it is in the interest of the licensee to accept the terms of the license. Such modified copies are called "forks".

Since any work on reproduceable media is automatically covered by copyright which prohibits distribution (Laurent 2004, 2), one needs to acquire a license in order to distribute copies. As a result, anyone who redistributes open source software must

consider himself to be a licensee of the software. Unless that person has been able to negotiate a different license with the original creators of the work, that person therefore relies on the rights granted by the given open source license and is therefore bound by it. Otherwise it would be unlicensed distribution which would be a copyright violation. The GPL even contains a provision that states that if the licensee violates the license in any part that entity automatically looses all rights granted in the license.

One key challenge for open source licenses is that they are expected to work in a globalized economy. This is somewhat alleviated by the fact that many countries have agreed to a base line with the Berne Convention (Laurent 2004, 151-152). None the less, there are differences in implied warranties which might be very different in each country or in how specific aspects of a license work. For example, the GPL makes it optional if newer revisions of the GPL automatically come into effect. This is not in accordance to German law under certain circumstances (ifrOSS 2005, 10). That being said there have been few legal battles over open source licenses. One of the few decision by a court so far as been in Munich Germany (ifrOSS 2005, 23). Other cases have been settled outside of courts, such as a dispute between MySQL AB and Nusphere, or are still ongoing, like the case between SCO and IBM.

## 3.4 Development Process

The open source development process has disproved, at least on the surface, key assumption about the organization of labor: "People do not easily work together in large groups toward a joint goal" (Weber 2004, 8). Raymond notes in reference to the well known book "The Mythical Man-Month": "Brook's law predicts that a project with thousands of contributors ought to be a flaky, unstable mess. Somehow the Linux community had beaten the $N^2$ effect and produced an OS of astonishingly high quality" (Raymond 2001, 171). This somewhat inaccurate representation of Brook's Law, which actually states: "Adding manpower to a late software project makes it later". It does however illustrate that it is an "oversimplification" to which the author admits (Brooks 1995, 25). For example, the law talks about project that are "late" which does not apply to the usually non-fixed release dates set in open source (Jones 2000).

The open source development process is considered to have a similar revolutionary impact as the publication of the "The Machine that Changed the World, a study

of the Toyota lean production system for manufacturing autos" (Weber 2004, 224). While this process has emerged from the open source community it is not intrinsically linked to it: "It is possible to apply open source collaboration principles inside a large company, even without the intention to release the resulting software to the outside world" (O'Reilly 2004). The key challenge is aligning the business and the community interests in such a way to create synergies which form the basis for collaboration.

### 3.4.1 Motivation

There are several aspects that signify this approach to software development: "We're used to thinking of software as an artifact rather than a process" (O'Reilly 2004). The open source approach changes this. More importantly, it allows anyone to participate in this process although not necessarily on equal terms. This includes fellow developers, administrators, end users, even competitors. In other words, open source development process is about "treating your users as co-developers" (Raymond 2001, 27). The basis on how much control one has over the process is determined by "a new kind of IP: reputation property instead of intellectual property" (Asay 2006*f*, 106). Furthermore, "in nonauthoritative settings - that is, in relationships primarily characterized as bargaining relationships - power derives in large part from asymmetrical interdependence" (Weber 2004, 229).

So what motivates developers and companies to incur the opportunity costs in developing source code essentially for free? Contrary to what might seem as the only sensible cause for people to give away things they could otherwise get paid for: "I do not see evidence for an explanation based on altruism, which (in some formulations, but of course not all) stands opposed to rational behavior" (Weber 2004, 226). That being said, studies have shown that open source developers nonetheless claim that altruism plays a role in their participation (Stoll 2006, 20). The political and ethical language used by the Free Software Foundation also seems to at least show that there are potentially non monetary interests involved. Lerner and Tirole however pose a valid question in this case: "Altruism has not played a major role in other industries, so it would have to be explained why individuals in the software industry are more altruistic than others" (Lerner & Tirole 2002). Michael Olson, the CEO of the recently by Oracle acquired database company Sleepycat, makes his motivations clear though: "I run a business based on open source, but my agenda is commercial, not political" (Olson 2006, 72).

Other important reasons cited point out that open source is a means for students to build up a reputation while they educate themselves (Stoll 2006, 20) (Lerner & Tirole 2002). This may be one of the reasons why open source licenses usually protect against removing of credits. Another reason for motivation also mentioned is the often strong identification that developers build up with the projects they are involved in due to the community approach in open source. This might be attributable to the fact that while open source is not free from hierarchy, actually being notably elitist (Lerner & Tirole 2002), that this hierarchy is derived from reputation acquired through contributions to the project. Further study would be necessary in this realm, but it would seem plausible that since reputation is given by the community, that people perceive this as a more natural hierarchy compared to companies where managers get installed in high positions without having to prove themselves to the workforce underneath them.

All of the previously mentioned points add to the fact that developers simply enjoy working on open source more than working on proprietary software (Stoll 2006). This might indicate why open source has traditionally struggled with finding people willing to work on "boring" tasks, also refereed to as "less glamorous" (Lerner & Tirole 2002) or "drudgery" (Fogel 2005, 25), like documentation or to accept limitations due to maintaining backwards compatibility (Lerner & Tirole 2002). None the less, several projects have managed to accumulate highly sophisticated manuals (Stellman & Greene 2006). While documentation is obviously very useful, in open source one can at least always turn to the source in order to find out how an application behaves. The issue of backwards compatibility as well as forking will be addressed in a later section.

Most projects, certainly the more successful, are started because of what Eric Raymond describes as "scratching a developer's personal itch" (Raymond 2001, 23). Similarly, another important reason for participating in open source is simply that one is also an end user of the same product: "The stupid network empowers innovators to try out their innovations without having to ask permission or procure a license from the network owner" (Weber 2004, 232). The term "stupid network" in this case refers to software production and distribution: "Put simply, end-to-end innovation systems depend on a market logic of disaggregated decisions rather than on a central intelligence to manage the innovator's dilemma" (Weber 2004, 237). Open source enables the end-users to prioritize modifications and to experiment with innovative approaches, rather than expecting a central authority in the form of a proprietary vendor to be able to guess what the market needs: "Empower people to experiment [..] Enable bits of information to find each other [..] Structure information so it can

recombine with other pieces of information [..] Create a governance system that sustains this process" (Weber 2004, 234).

### 3.4.2 Structure

In order for information to be able to easily "recombine with other pieces of information" open source development has put particular emphasis on modularization: "By far the most common characteristic open source projects have is a highly modular design" (DiBona, Cooper & Stone 2006, xxviii). This makes it possible for anyone to easily replace parts of a working system with alternatives that more closely match their needs: "Distributing source code and licensing it under GPL ensures low and nondiscriminatory barriers to entry. The decision about when and how to innovate then lies with the user on the edge of the network" (Weber 2004, 233). Looking at the big picture it does more however: "It enables parallel processing of a complex task in a way that is not only geographically dispersed but also functionally dispersed" (Weber 2004, 233).

As a result, open source projects often end up being structured as many sub-projects within a single larger community. This is one of the key elements that assist in fighting the issues discussed in Brook's law. For example, new members are not necessarily added to existing sub-projects when they first join as they can often simply create their own sub-project to add whatever features they felt were missing. Due to modularization it no longer becomes necessary for all the members of the community to discuss all aspects of the software. This counters the significance of the exponential growth of communication paths. Studies have found indications of this when they were analyzing the communication paths for small to large sized projects (Crownston & Howison 2004). Their data indicates that communication is actually centralized and that this centralization is maintained by the fact that larger projects organize themselves as several sub-projects. At the same time, each project or sub-project is often structured similar to an onion:

> "At the center of the onion are the core developers, who contribute the most of the code and oversee the design and evolution of the project. In the next ring out are the co-developers who submit patches (e.g. bug fixes) which are reviewed and checked in by the core developers. Further out are the active users who do not contribute code but provide use-cases and bug-reports as well as testing new releases." (Crownston & Howison 2004)

It should also be noted that open source projects vary greatly in size. Studies have shown that the bulk of projects are very small. For example, a study of the 100 most active projects on the popular Sourceforge.net open source development repository showed that "the median number of developers [..] was 4 and the mode was 1" (Krishnamurthy 2002). Successful projects are also able to attract more developers, likely due to their increased signaling ability (Krishnamurthy 2002) (Lerner & Tirole 2002) which helps in focusing resources.

The large numbers of testers may be one of the reasons why open source code has been shown repeatedly to have less bugs than comparable proprietary code (Lemos 2004) and that more popular larger projects also tend to have less bugs (Chelf 2006). However, this may also be attributable to the fact that due to the open nature of the code people feel more pushed toward keeping their code as clean as possible as contrary to proprietary code it can actually be evaluated by outsiders. It is common practice for open source projects to send all changes to the code to a dedicated mailing list to facilitate peer review. Eric Raymond formulates the "Linus Law" based on this observation: "Given enough eyeballs, all bugs are shallow" (Raymond 2001, 30). Linus Torvald however makes the point that he would characterize the situation as follows: "Somebody finds the problem and somebody else understands it. And I'll go on record as saying that finding is the bigger challenge" (Raymond 2001, 30). Ben Laurie also notes that "although it's still often used as an argument, it seems quite clear to me that the 'many eyes' argument, when applied to security, is not true [..] Once you have found a bug, many eyes will, and indeed, do, make fixing it quick and easy" (Laurie 2006, 60). At the same time, the argument that open source makes it easier for hackers does not hold true in his eyes given the availability of excellent tools like IDA (a very capable disassembler) and Ollydbg (a power [and free] debugger)" (Laurie 2006, 65) to reverse engineer proprietary code.

Laurie goes on to define that "software is reliable if it does what is expected when operated as expected. It is secure if it does what is expected under all circumstances. This is a very critical difference, indeed" (Laurie 2006, 60). Finding all of these circumstances where the behavior is unexpected and feeding this information with as much detail as possible to developers is however where the "many eyes" principle shines. Furthermore, the ability to determine risk and to be able to prioritize the level of risk someone is willing to take is a key principle in security (Laurie 2006, 65). Having the source code in your hands and being able to build the binaries yourself is a must in this case. It would not be sufficient to be able to read the source code while using a third party binary since there would be no way to ensure that the source code really does match the binary in this situation. Ben Laurie therefore concludes: "It seems to

me there's little to be sensibly said that, from the viewpoint of security, truly differentiates open and closed source. The points I believe are critical are my ability to review the code and my ability to fix it myself when it is broken" (Laurie 2006, 70).

### 3.4.3 Release Management

It is quite common for open source software to provide nightly builds to its users. This is the ultimate in bleeding edge. However, open source software is also known to even make proper releases quite often. In order to shorten feedback loops between the developer and end user community open source is also known to follow the "release early, release often" (Raymond 2001, 28) mantra. Studies have shown that open source is faster in fixing bugs than closed source (Reavis 2000) (Challet & Du 2005). But even the normal road map often calls for releases every few months. Obviously the fact that most open source software is not sold in retail boxes in stores, but instead relies on the Internet to bring both the code and documentation to its users is a key enabler here. The short release cycles are also likely to motivate more contributions, since contributors know that the features they have added have a chance of being released within their own project timespan.

Open source software is also often said to break backwards compatibility more often (Lerner & Tirole 2002), as backwards compatibility is often considered to be a hindrance in order to streamline or even radically improve an existing product. Maintaining compatibility with an older API often requires additional resources during development or may even slow down an application. At the same time, open source projects often maintain multiple so called "release branches". For example, PostgreSQL still publishes updates to its 7.4 and 8.0 branches next to its current 8.1 release branch. Since the source code is open, users are also free to back port changes. For example, when the Mozilla project decided to drop its monolithic browser and email solution in favor of Firefox and Thunderbird, a number of users joined to pick up the maintenance under the SeaMonkey project.

In some cases developers might not agree on how to best approach future developments. Some might prefer a more conservative approach while others feel that they want to do some more radical innovations: "One common reason that open source projects die is that the community behind the project has a schism and some of the developers copy the source code to a new repository and start doing things their own

way. This is known as forking the project" (Woods & Guliani 2005, 16). The popular SAMBA project for example split up in a situation like this although this was not even considered to be a bad thing by leading members of the existing project (Tridgell 2000). While this may result in splitting up development resources, it ensures that at least end users end up with a choice. It also shows that contrary to the "chasm" explained toward the end of the last chapter, there might be a place for innovators in open source that continue to push for radical innovation parallel to providing a stable product. Microsoft ran an ad about this in relation to Linux (Figure


Figure 3.1: Microsoft anti Linux advertisement: "an open operating system doesn't only have advantages"

3.1). The idea of the advertisement seemed to be to illustrate that forking would lead to a confusing number of variations. The open source community however was quite oblivious to this criticism and rather felt that it was a good portrait of how open

source is able to adapt in the spirit of evolution and natural selection (Wolf 2000). Similarly, even if all the original developers decide to leave a project, end users can always decide to pick up the development: "In some respect, most open source projects never really die" (Woods & Guliani 2005, 15). This is important since this makes it worthwhile for companies to allocate development resources to open source project, that they rely heavily on in order to be able to react to and even directly influence the development (Woods & Guliani 2005).

### 3.4.4 Proprietary Development

While some aspects in open source development obviously require that the source code is actually open for people to modify, a lot of aspects could also be used in proprietary development. An example of this is "HP's printer division, where CollabNet's SourceCast platform is used to help more than 3000 internal developers share their code" (O'Reilly 2004). As a matter of fact, CollabNet has made it their business to provide consulting in how to best bring open source development principles into established companies. However, traditionally proprietary development is organized in a more hierarchical manner with managers at the top and developers underneath. The common argument here is efficiency. However, "no one says that hierarchical coordination in a complex production task like software development is efficient, only that it is less inefficient than the alternatives" (Weber 2004, 10). Efficiency might still not make you successful though. Just as homogeneous communities might be very efficient but die out quickly when a new disease appears: "Hierarchies have a difficult time fighting networks, it takes networks to fight networks, and whoever masters the network form first and best will gain major advantages" (Weber 2004, 263). That being said, "open source, by itself, is as much a losing strategy as closed source, by itself" (Asay 2006*f*, 115). Open source relies on third party packagers. For example, most people rely on Linux distributions to put together a complete operating system environment instead of assembling their own setup around the raw Linux kernel. Yet, due to their at least partial disconnected nature, "a recent Forrester Research report claims that packagers are actually quite slow - as slow or slower than closed source companies - at getting out fixes" (Laurie 2006, 64). Open source is not the "silver bullet".

## 3.5 Commoditization

Tim O'Reilly defines commodities as follows: "Commodities are always sourced by more than one producer, and consumers may substitute one producer's product for another's with impunity" (O'Reilly 2004). Commoditization is therefore the process by which an technology, that was previously exclusively produced by one company, becomes available from multiple sources. The word commoditization is often considered a bad word. It stands for devaluation of previously high end technology: "One might be tempted to see only the devaluation of something that was once a locus of enormous value" (O'Reilly 2004). Even worse: "The forces of commoditization, being natural market forces, cannot be beaten. Yet time and time again, incumbent firms fight them. [..] In very simple terms, this is Clayton Christensen's Innovators Dilemma at work" (Murdock 2006, 92). However, this process produces opportunities: "To use Schumpeter's phrasing, it is easier to see precisely the destructive side of creative destruction, than it is to see the creative side" (Weber 2004, 15). That being said, the "choice" technology companies are faced with seems to boil down to the following: "One either commodifies or evades commodification through innovation" (Asay 2006*f*, 114). This is due to the fact that "even software that starts out proprietary eventually becomes standardized and ultimately commodified" (O'Reilly 2004).

Standardization is one of the key aspects here: "The moral of the story is that standardization, and thus commoditization, are both natural market forces as well as key events in human history" (Murdock 2006, 92) while "open source hastens software's natural trend toward standardization/commodification" (Asay 2006*f*, 103). Market commoditization reduces the ability of vendors in this market to keep their customers locked-in due to reduced switching costs. Conversely, software companies that have a manufacturing and distribution advantage or that are aiming to sell complimentary products or services have an inherit interest in commoditizing a given market. Feature innovation such a market is obviously not of great importance anymore. The open source development process therefore creates an opportunity to chip away at proprietary competitors, pushing them up-market (Asay 2006*f*, 104). Mark Olson is sure to point out here that "this is not to say that open source development is not innovative, but its impact has been largest in the cases where it has commoditized products in mature markets" (Olson 2006, 90). The resulting devaluation is likely the origin of the "open source is communism" rhetoric, but this is far from the truth: "A historical view tells us that the commoditization of older technologies and the crystallization of value in new technologies is part of a process that advances the industry

and creates more value for all" (O'Reilly 2004). As a matter of fact is that capitalism does not care for established companies.

Microsoft is one of the first companies to master commoditization in the software world: "By offering its products at lower prices than its competitors could afford to offer them, Microsoft preemptively commoditized many of the markets in which it competed, depending on high volume to make its products profitable and making it impossible for challengers to undercut it." (Murdock 2006, 95). Just like IBM handed Microsoft the opportunity to become the largest software producer by letting them provide the operating system for their PC standard, they left the reaping the benefits of this standardization on the hardware-front to the manufacturer Dell (Murdock 2006, 96). Companies must realize that "operating in a commodity market calls for entirely different business models than the business models that have preceded them" (Murdock 2006, 97). Obviously both Microsoft and Dell are reaping in huge profits in their respective markets. The same applies to other previously commodified markets: "Virtually all companies today continue to spend heavily on electricity and phone service, for example, and many manufacturers continue to spend a lot on rail transport." (Murdock 2006, 102).

## 3.6 Productization

Productization is the process of turning a piece of technology into a product that end users can put to use. This usually entails making sure that the product works in various environments and comes with the necessary documentation and packaging: "The fact is that most software, open source and commercial, has problems that must be overcome for it to be useful" (Woods & Guliani 2005, 24). More importantly, it is also about making sure that the given technology is assembled in a way to solve a problem the market demands a solution for. This is a non trivial challenge for any company: "Productization requires a huge amount of work. It can take double or triple the amount of work it took to complete the original features and turn a program into a product" (Woods & Guliani 2005, 21). This applies to both proprietary as well as open source software (Woods & Guliani 2005, 24). However, traditionally open source has put less emphasis on productization: "The technical aesthetic that the open source programmer depends on is more focused on solving complex programming problems than it is on engineering for convenient user interfaces and simplicity" (Weber 2004, 238). This may be attributed to the fact that since in open

source the original developers are often the main users, they essentially evade some of the reason that push the idea of productization: "At the core of the creative process for commercial software is a vision that many people will pay for the software being created. For open source software, the intended audience and the developer are usually the same. So there is no mystery about what the audience wants" (Woods & Guliani 2005, 19).

Open source projects for which nobody does the necessary work in productization will find that there is a natural limit of growth within their community. Linux distributions were undoubtedly critical for its success. But even here we notice that Linux has not managed to make a significant dent into the home PC market which arguably requires the most sophisticated productization. More over it can be noted that "it is no accident that the most skilled engineering teams in the country are also the largest users of open source. For them, the cost of overcoming the productization gap is small" (Woods & Guliani 2005, 23). The creation of "the killer applications of the Internet era", in the form of "Yahoo!, Google, Amazon, eBay", based on open source technology was therefore left to technology-oriented companies that build up the IT know-how internally to exploit the benefits of the open source driven process of commoditization (O'Reilly 2004).

These companies require a high level of customization, the ability to grow quickly without having to hand over increasing license fees. This makes the investment into the required know-how worthwhile, which might not make sense for smaller companies. On the other hand, smaller proprietary software customers have other issues. They have no strength to fight against steep license fees due to vendor lock-in. They have no control over when issues get fixed. Especially for edge users of popular software, customers may find it hard to motivate their vendor to cut a deal, add a niche feature or fix a tricky issue. The situation might even become hopeless when the vendor goes out of business or the product is pulled after being bought by a competitor: "In contrast, with open source, users are never at the mercy of the maintainer. They can always fix the problem themselves" (Laurie 2006, 65). So it seems obvious that there is also an opportunity for companies to provide productized open source and related services to companies. As such, the lack of productization is actually a business opportunity.

## 3.7 Business Models

Aside from the technical challenges associated with productization there is also the question of what business model to choose. Some even say that the "most important innovations for the next decade of software will come from business model innovation" (Asay 2006*f*, 103). Open source is said to play a key role here (Onetti & Capobianco 2005). One approach for software companies has been to sell licenses for a set of core products, another to sell complements such as support, training and add-ons. A surprising fact may be that a study of the U.S. Census Bureau showed that in 1997 only 24.6% of the software industry makes its money with "prepackaged software" (OTEC 2006). That was even before open source became big with the Dot-Com boom. This means that the large majority of software is not sold as off the shelf software. The same study showed that 19% of turnaround is generated from "computer programming services". Consulting, data processing and other services are the only other areas to get into double-digit percentages.

This clearly shows that there is a lot of market potential for companies implementing software rather than selling licenses; a great opportunity for open source software if it can provide solid products to the market:

> "One ideal that is rarely achieved is to merge the creativity and technical brilliance of the open source world with the operational discipline and process of the IT. But the two sides look at each other with disdain. The open source experts look at IT and see a massive skills gap: what is so hard about picking up and maintaining the skills needed to use open source? The IT professionals look at open source software and frequently see a productization gap because of half-finished products: what is so hard about finishing all the administrative interfaces, configuration tools, Application Programming Interfaces (APIs), and documentation to make the software useful?" (Woods & Guliani 2005, 4)

Open source may also be relevant in situations where the main goal is not to make money with the given software, but instead to reduce costs inside the internal IT infrastructure. Any company needs software these days to operate efficiently. Starting from managing liquidity and the workforce, to knowledge management and customer communication, software plays a critical role. The key question is if a given piece of software is non-differentiating or not (Perens 2005). Open source advocate Bruce Perens estimates that "90% of the software in any business is non-differentiating" (Perens 2005). For example, choosing an operating system is usually

a non-differentiating feature. Customers rarely care if the emails they receive have been written on a computer running the Windows or the Linux operating system. That obviously does not make that choice irrelevant, as this choice could determine the internal costs which factor into the final cost associated with the product that is being provided for the customer. On the contrary, differentiating software is what sets a company apart from its competitors. For example, Google's search algorithms can be considered as one of their key differentiating advantage over Yahoo! and MSN.

Even this example is not clear cut. Even with the algorithms open sourced there is still more to Google, like the exact organization of its infrastructure, its employees and other significant factors. Obviously, one needs to build a large enough community to be able to leverage the benefits mentioned in the previous section on the open source development process in order to gain from open source. Once this daunting task is achieved, there are more challenges to meet: "If the company alienates the open source community, the main advantages of open source distribution - ubiquity and large installed base - can disappear in a flood of bad press and ill will on developers discussion lists" (Olson 2006, 86). A prediction that still needs to truly validate itself states that "as the IT industry begins to recognize the promise of emerging open source business models, community becomes less critical to the success of a project" (Asay 2006*f*, 108). Either way, the situation seems to rest on variables which are hard to control or in other words: "If measuring the parameters were easy, the strategic choice to open source would be a function of comparing benefits you gain from the distributed development effort to costs you incur getting it started along with savings you will bring to your competitors by giving them free access to the same tools that you have" (Weber 2004, 266).

At the core of all of this is how to best manage intellectual property rights: "In both the proprietary and open source cases, the owner grants licenses to use the software under certain conditions" (Olson 2006, 75). Note that even proprietary software can be distributed free of charge known as freeware and shareware. Here too, the goal can be to increase the market for complementary products or even to kill profitability for a more successful product from a competitor. While proprietary vendors must invest into technology to prevent users from stealing their software, often times adding discomfort for their paying customers along the way, open source vendors are naturally immune to software piracy. That being said, "if you give your product away, anticipating juicy follow-on sales based on consumer loyalty/switching costs, you are in for a rude surprise if those switching costs turn out to be modest" (Shapiro & Varian 1999, 149).

Similarly, "open source needs to interact, deeply and efficiently, with existing structures of capitalist economies and legal structures like copyright, patent law, and licensing." (Weber 2004, 18). Open source has shown that some of these perceived boundaries do not hold true when put to the test with open source which is able to solve some of the issues previously accepted as inescapable (Weber 2004, 231). Put into more extreme words: "My argument here simply is that shifting property rights can and will likely destabilize the foundations of existing cooperative arrangements and institutions, and possibly in a more radical ways than do changing transaction costs" (Weber 2004, 257). Generalized advice tends to be dangerous but when taken with a grain of salt it can be helpful: "Companies should adopt the both source strategy when they are on top of their games, instead of when they are losing the final sets of their matches. For market losers, a better bet is to make the difficult transition into a pure-play open source vendor" (Asay 2006*f*, 115).

Varian agrees that IP is a key question of any business: "The challenge of intellectual property management lies in trading off these two effects: in choosing the terms and conditions that maximize the value of your property. The more generous the terms on which you offer your intellectual property, the more you can charge, but the less you sell" (Shapiro & Varian 1999, 98). However, the intriguing aspect of open source is that it flips this upside down. Also, "the fact that many users are able to use the software at no charge actually takes money away from purely proprietary vendors in the market" (Olson 2006, 78), which creates sort of a double whammy for proprietary competition. At the same time, it provides for very generous terms along with effectively zero license fees. Customers therefore are increasingly attracted to open source: "Companies and governments that buy large software packages for long-term use increasingly value this aspect of open source standards because it inoculates them against later exploitation of lock-in by a dominant supplier" (Weber 2004, 239). As previously mentioned, customers also have the freedom to fix issues and assess security risks with open source, which enables customers to choose open source based software stacks over proprietary alternatives. Companies that make their money with solutions based on these open stacks obviously stand to gain in this case.

But open source companies also benefit from lower marketing and sales costs as well (DiBona et al. 2006, xxxiv). Looking at the income statements of Microsoft, we can see that in the third quarter 2003 and 2004 research and development had similar costs as sales and marketing (SEC 2004). For the first quarter of 2005 and 2006 respectively Oracle filed that only 14% of its expenses go towards research and development while 22% went into sales and marketing (Oracle 2006). Open source on the other hand can lower these marketing and sales costs as customers can simply download and run it

without the hassle of dealing with a vendor specific license. Furthermore, in-depth review all the way down to the source code is possible. As a result, "open source has very low marketing costs that create an inbound sales channel of prequalified leads" (DiBona et al. 2006, xxxv).

However, due to the relative young age of this phenomenon it is not yet researchable. Open source companies will likely at some point find that there is a natural barrier that they can only overcome if they employ a direct market force (Hilf & Asay 2006). The fact remains though, that in the early days of existence in which companies must aim to keep expenses low, they do not need to spend as much for marketing and their own software infrastructure. Misinvestments into proprietary software that fails to meet the requirements can also rarely be recovered, which is an important consideration for customers (Perens 2005). This leads to the problems Dave Dargo, Ingres CTO and long time Oracle employee (Ingres 2006*a*), labeled "the broken covenant of software licenses" (Dargo 2006*c*). He argues that in a mature market paying up-front license fees in order to get new features makes no sense. In that way he proclaims that open source is nothing more than the market fixing this imbalance. Similarly, users will eventually realize that their vendors are effectively charging them money for upgrades they will never install (Dargo 2006*a*). More importantly, customers may realize that without the offer of new useful feature updates, vendors effectively only have the option of trapping their customers in order to still be able to exert license fees (Dargo 2006*b*). In that sense investing into time-limited support rather than perpetual proprietary licenses is very attractive and as a result companies offering products build around support fees could only benefit from this market demand.

As previously mentioned, academic research relies on being able to reproduce and extend other peoples work. This makes open source attractive for students and professors alike. Moreover, students will eventually become decision makers once they finish their degrees. They are likely to prefer tools and concepts they are already familiar with (Shapiro & Varian 1999, 47). Open source products also tend to induce users to very active advocacy. While it's not uncommon for proprietary products to benefit from end user support websites, open source takes this somewhat further. As an extreme example of this, Spread Firefox was a community driven marketing effort that managed to collect sufficient funds to run a two page add in the New York times (Baker 2006, 16).

The potential for savings are also to be found in the product development itself. Getting free developers, free bug fixers or even highly motivated bug reporters that provide in-depth analysis is obviously an attractive proposition. This is especially il-

lustrated by the open sourcing of Netscape to form the Mozilla project. Back then "the combination of open source techniques with an active, focused commercial management structure was unchartered territory" (Baker 2006, 3). With this change "the percentage of mozilla.org staff employed by Netscape decreased steadily. Today, the mozilla.org staff does not have any Netscape/AOL employees" (Baker 2006, 6). This was a non trivial undertaking:

> "Suddenly the decision of what code goes into the source repository is not made by managers for a particular person through the employment process. Instead, the decision is made by engineers who review the code itself rather than the person's credibility or employment status [..] Netscape's role was now determined by classic open source principles: leadership and influence through the quality of one's contribution." (Baker 2006, 9,11)

Firefox, based on Mozilla code, is the only other browser next to Microsoft's Internet Explorer that is able to be in the double digits in web site views (OneStat.com 2006) (Net Applications 2006).

### 3.7.1 Support

The most obvious business opportunity for open source companies are services like support, consulting, training and certification. Reducing the costs for customers to start using a complementary product via commoditization makes a lot of sense from a business perspective (DiBona et al. 2006, xxxiv). In this way, the software can be considered a complement to the support offerings. However, there are two challenges here. First "selling support or services, by contrast, imposes new costs with every deal" (Olson 2006, 73). One way out of this dilemma is certification and support guarantees as illustrated by Red Hat:

> "More precisely, customers were free to redistribute Enterprise Linux, but in doing so, they lost all support from Red Hat and, most importantly, from the legions of ISV and IHV's that certified to the Red Hat platform. [..] Red Hat's new model was still in keeping with the letter of the open source movement but no longer with its spirit." (Murdock 2006, 100)

The second issue is that "while it is true that the original authors of an open source package have an advantage in supporting it - after all, they know the most about how

it works - freely available source code permits anyone else to learn the product internals and offer the same service" (Olson 2006, 81). Recently, a number of companies have formed around the business model of assembling existing open source products for their clients (Marshall 2006). These companies try to keep custom code development to a minimum and instead try to leverage existing open source code as much as possible to reduce costs and project lead times. This kind of approach follows well with Gartner's concept of agile development (Schindler 2006). But even companies who are involved with only one open source product can also make money from custom software development. They will usually give their customers the option of making the custom code open or not, potentially offering reduced fees as an incentive. But customers should realize that any changes they choose to not open, may incur additional costs down the road as the custom code may need to be adapted over and over again to maintain compatibility with the open code base.

### 3.7.2 Dual-Licensing

Another business model that has produced such highly acclaimed open source companies like Trolltech, JBoss, Sleepycat and MySQL AB is known as "dual-licensing". In this scheme businesses "do not use collaborative development to build their products" (Olson 2006, 73) and instead "rely on open source as a distribution strategy, not as a production strategy" (Olson 2006, 79). Depending on the exact approach there will be no or only very few external developers: "JBoss, for its part, employs roughly 85% of the developers who contribute to the JBoss open source project" (Asay 2006*f*, 115). The key aspect with dual-licensing is that the given company maintains the full IP over the software. This enables these companies to release the software under an open source license while also selling proprietary licenses: "The open source license must cause enough pain that some users would rather pay money than endure the pain" (Olson 2006, 83). The license of choice for this is usually the GPL which due to its propagating nature requires that all code that is linked to it also be licensed under the GPL. The "dual-licensing strategy provides the customers with a mechanism to buy their way out of the GPL" (Asay 2006*f*, 116) and therefore enables them to not be bound to the GPL. This is essentially the concept of versioning. The GPL licensed code is for companies willing to comply to the GPL thereby "repaying" the original producer by enhancing the ecosystem around the product. Other companies have to pay for the license if they choose the proprietary version.

This obviously means that these businesses lose the benefit of getting significant code

contributions from the outside, since these cannot be accepted unless the contributor also signs over the IP for the code. However, due to the open nature of the code, they may still benefit from an active community that helps in finding bugs and educating new users. This also means that customers must rely on the same tactics as proprietary vendors to get control over the development process which is to waive money if they want something changed. In more traditional open source they could alternatively hire a core developer to influence the direction of the project. However, the marketing and sales advantages still apply: "Dual-licensing businesses can distribute software to more people, more cheaply, than their proprietary competitors" (Olson 2006, 73). Customers who do not mind being bound to the GPL, because they do not distribute software, do not need to acquire a license. This can be considered a variation of "third-degree price discrimination" (Shapiro & Varian 1999, 44).

### 3.7.3 Proprietary-Extension

A final approach that can be taken with open source code under a non-propagation license is proprietary-extension. Since these licenses allow closed source distribution of modified versions, software companies can take code from these projects as a foundation for their own closed source products. Again this makes sense mostly if only the non-differentiating components are open source, since non-propagating licenses would allow any customer to freely use the product as well. For example, although Microsoft never confirmed the rumor of using the TCP/IP network stack from the open source FreeBSD project (Lehey 2001), they do admit to using open source (Gomes 2001). In this business model companies can use all the collaborative advantages of open source for their core product and partially also benefit from marketing due to their close association with the underlying community.

## 3.8 Impact on Society

Although the free software camp does have the stated mission statement of replacing all proprietary software with free alternatives, it does not mean that there is not room for both to exist side by side: "Dual-licensing will never replace either pure proprietary or pure open source strategies. There are compelling reasons for both to exist" (Olson 2006, 89). This obviously also applies for other open source business models as well. After all both are implemented with licenses in the same copyright law. In this

context it is also important to remember again the distinction between commercial and proprietary software. Commercial software can be either open source or closed source, while proprietary software means it will be closed source. At the same time, neither open source nor proprietary software is commercial by definition. Just as a lot of open source code is freely available on the Internet, there is closed source freeware just as readily available.

Even though this paper focuses on the benefits of parties participating in the open source process, it is nonetheless important to put things in perspective, so that policy makers can better weigh the impact of open source. In this context it should be mentioned that there are also disadvantages or at least limitations in the open source process. Due to the fact that open source is still available even if it's unmaintained, it may eventually become quite a burden to afford the search costs in finding a working and maintained open source solution. Sourceforge.net has introduced metrics that should help users overcome this challenge, however. Automatic code analysis tools help assess quality to some extend, too. A bigger threat therefore might result from the following observation: "If the resources cannot be depleted (for example, a digitally encoded symphony) but need to be provided by voluntary human action in the first place, a related tragedy can develop if nonexcludability means that no one has an incentive to provide the common good to start" (Weber 2004, 243).

The question if open source is able to provide sustainable innovation is beyond the scope of this paper. The question if open source is able to innovate can however be answered: Yes, it can. The best proof of this is that open source has been the driving force behind arguably the most significant computer related innovation in the recent decades: the Internet. While commercial companies were unable to produce a business strategy around the Internet which for example led NCSA to drop their web server project, it was open source that picked up where commercial companies have failed (O'Reilly 2004). The result was the creation of the most popular web server, Apache. The Internet relies on open source solution such as BIND and Sendmail as its backbone software infrastructure for DNS resolution and e-mail traffic (O'Reilly 2004). Both have spawn commercial companies for related services.

Mark Olson notes that maybe even common "wisdoms" about innovation must be questioned or at least revalidated at times: "The amount of innovation in the software market is much smaller than is generally supposed" (Olson 2006, 77). A discussion of which development method, open or closed source, is able to be more innovative cannot be determined in the context of this paper. The point is that open source is able to innovate and as a result is at least a viable alternative that warrants further

study. It could also be that the most significant innovation of open source is not in the software itself. Microsoft conducted a study that showed that almost all end users never expect to modify the source code, yet still over half considers access to the source code critical (Asay 2006*f*, 106-107). This illustrates that "source code matters because it shifts control back to the buyer, which forces vendors to offer better software at lower prices" (Asay 2006*f*, 107). The loss of resources due to vendor lock-in (Asay 2006*f*, 111) and discontinued products (Woods & Guliani 2005, 22) should not be underestimated. Open source "functions as a security [..] if Vendor X fails to deliver on its promises, or if it goes out of business, that customer will have the option (unpleasant though it might be) to have some other service firm support stranded code" (Asay 2006*f*, 110). In slightly simpler words: "IT that works for a customer, rather than against it. That is innovative" (Asay 2006*f*, 112).

Steve Weber has examined open source from the perception of a political scientist. He observes that the fact of the matter is that the current IP regime is nothing more than a "pragmatic compromise" (Weber 2004, 230) that has to be reexamined: "Move the boundaries and some things will not be produced. Some incentives will unravel. Something will be lost. On the other hand, something will be gained as well" (Weber 2004, 248). Open source licenses have made it possible to work within the old regime while still providing the legal basis for an entirely different view on IP rights management. Weber notes that "the core problem of intellectual property is supposed to be about creating incentives for innovation" (Weber 2004, 4). As such, he is "not making the argument that current intellectual property is 'wrong'. it is not wrong, but neither is it right. It simply modulates incentives to produce and distribute things in particular ways" (Weber 2004, 247). On the other hand, "open source is an experiment in social organization for production around a distinctive notion of property" (Weber 2004, 16) which relies on expanding the commons to bring value to the society as a whole.

Open source does require a slight transformation in how businesses approach software however: "Perhaps the biggest benefit of using open source is the process of self-improvement that an IT department undergoes in learning how to make open source work" (Woods & Guliani 2005, XI). The benefits have already been mentioned: less danger of vendor lock-in with reduced risk when products are no longer maintained by the given vendor, opportunity of distributed parallel development driven directly by end users, the opportunity to better assess security risks and many more. At the same time, open source does not preclude crucial services (Woods & Guliani 2005, 22) such as support: "For good reasons, managers like the idea of one throat to choke if something goes wrong" (Woods & Guliani 2005, 3). Quite the contrary, support is one

of the main sources of income for open source companies. So for the most part customer can easily choose to ignore the fact that they are using an open source product if they rely on a vendor to implement the software for them, while benefiting from reduced fees coming out of the reduction of transaction costs.

From the perspective of a global community open source is even more significant: "From an efficiency perspective, the possible exclusion of 4 billion people from the next era of wealth creation makes no sense. From an ethical standpoint, it is even more problematic" (Weber 2004, 249). The stated goal of IP rights is to facilitate innovation and if suddenly there is an opportunity to integrate the developing nations into this process, it seems to be a too great of an opportunity to pass by: "The promise inherent in this argument is that software innovations can and should come from everywhere. Emerging markets are not implicitly stuck relying on commoditized, hand-me-down innovation from the developed world" (Weber 2004, 252). Again the point is that developers in these regions will likely have an entirely different angle to their immediate needs, which means there is an opportunity to get innovative solutions from these regions. It seem that developing nations are very eager to grasp this opportunity they are getting through commoditized open source software (DiBona et al. 2006, xxxvi).

However, just like developing nations, even western nations have found themselves dependent on other nation's software for mission critical aspects of their IT infrastructure. Governments around the world largely depended on Microsoft Windows as their operating system which is owned by the American company Microsoft. Google has increasingly been dominating the search engine market. Google and its two primary alternatives Yahoo and MSN are also American companies. Governments are realizing that they need to make sure that there are alternatives out there. For example, when Google decided to scan books in order to preserve them, there was an uproar in France: "I am not anti-American, far from it [..] But what I don't want is everything reflected in an American mirror. When it comes to presenting digitized books on the Web, we want to make our choice with our own criteria" (Riding 2005). Open source is a way out of this dilemma. Governments can influence a given project by making sure enough local developers are working on a given project. More importantly, they can also choose to fork the project if they feel the direction is no longer compatible with their national agenda.

The process associated with open source might even be applicable to other fields of study: "If the open source process has more general characteristics, if it is a generic production process for knowledge that can and will spread beyond software per se,

then the implications might be considerably larger" (Weber 2004, 264). There are already examples of this. For example, there is a project that aims to develop an open source graphic card design (Hancock 2006). Recently, a full length open source animated movie was released that was rendered using open source tools (Elephants Dream 2006). Wikipedia has become a significant player in the encyclopedia market while also being based on the open source principles. Maybe the key realization here is that in the information age it simply makes more sense to prefer open approaches over closed ones: "In all three cases, old-fashioned ideas, like news distribution, journal publication, and product reviews, have been transformed by the open, collaborative processes that the Internet encourages. Licensing is as much an issue here as in software distribution - ownership of the content, and the right to distribute it online, must be considered carefully" (Olson 2006, 89). Another example gives some more food for thought: "From the perspective of the medical care system as a whole there is enormous wasted effort when the doctor in New York ends up going through a problem-solving process that a doctor in Berkeley has already figured out in a very similar case, simply because she has no way to search for and access that knowledge efficiently - even though the California doctor has no reason not to share it (Weber 2004, 269).

# Chapter 4

# Relational Database Market

## 4.1 Introduction

Two research projects, Ingres at University of Berkeley and System R at IBM, were among the first implementations of Edgar Codd's relational data model (NAP 1999). The relational database market later standardized on using SQL as the language of choice for interaction with these RDBMS (Elmasri & Navathe 2000). The first commercial RDBMS were derived from either the Ingres or System R research projects. Missing from the timeline in figure 4.1 are several important embedded RDBMS such as Berkeley DB[1], Apache Derby and SQLite. Interestingly all three are open source databases although there are of course also proprietary offerings in the embedded market space. Key points to note here are that the market for RDBMS is very mature (McClure 1997) and open source seems to be able to compete well in this market (LaMonica 2005). The following two chapters make use of jargon established within the database community. Please refer to the glossary at the end of this paper for brief explanations for some of these terms. For information about the different databases in this market please refer to the end of this chapter.

## 4.2 SQL Standard and Future Trends

The SQL standard is maintained by the American National Standards Institute (ANSI) and more specifically by the ANSI-H2 committee. International members are allowed however, even if the name would seem to imply otherwise. Participation in the ANSI

[1] Berkeley DB provides an optional relational interface.

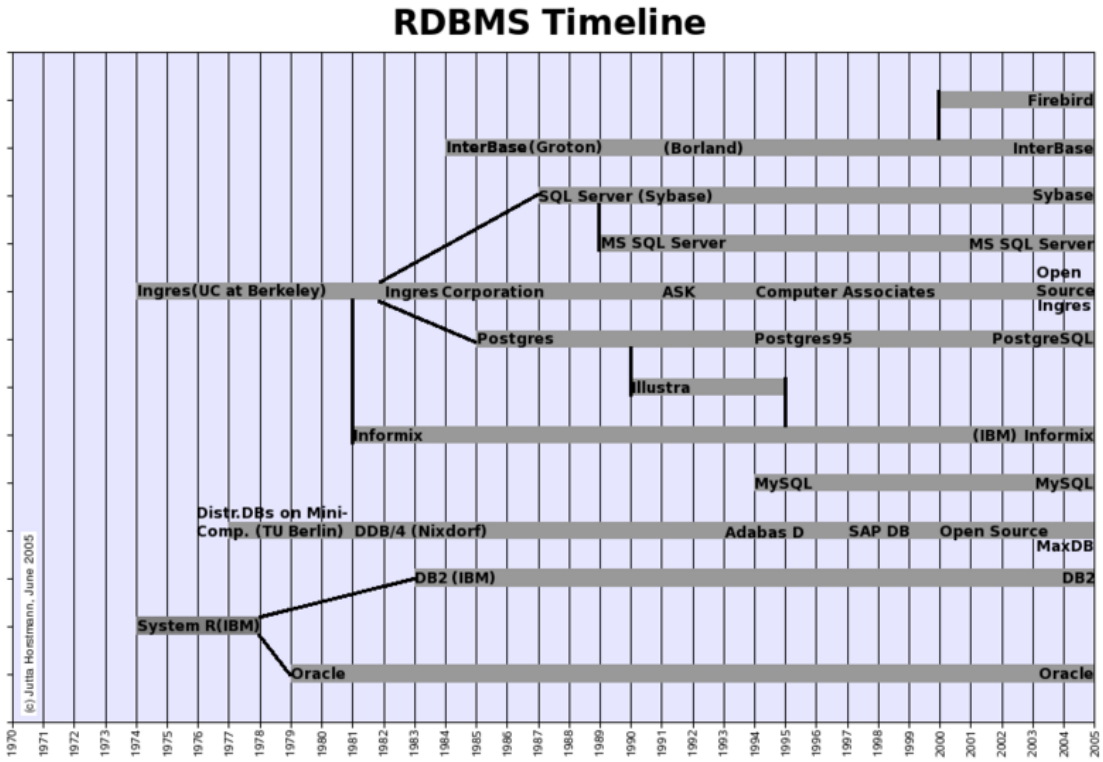


Figure 4.1: RDBMS timeline (Horstmann 2006)

is possible through becoming a member, which requires paying a membership fee that starts at 500 USD for individuals per year (ANSI 2006). Membership includes the ability to influence the definition of the standard. According to Michael Gordman, long time secretary of ANSI Database Languages Committee, the following SQL vendors were represented in ANSI-H2: IBM, Oracle, Sybase, Informix (acquired by IBM), NCR, Computer Associates (have since sold their Ingres SQL database), Microsoft, FileTek, InterSystems, Compaq, Pervasive (Gorman 2001). According to an informal survey (OSDB Consortium 2006) and a number of email interviews conducted for this paper (Appendix A) none of the open source projects questioned said it has any ability to influence the standard, even though several of the committee members are representatives from companies that also work on open source databases. Interestingly however, the example of the LIMIT clause and the REPLACE statement illustrate an informal standardization process within the open source community. The LIMIT clause was initially added to MySQL based on user feedback from Rasmus

Lerdorf. While equivalent behavior is available in many databases, this new syntax is much simpler and more explicit. Since its addition to MySQL, LIMIT has made its way into SQLite, PostgreSQL and Firebird[2], all of which are open source databases. Similarly, the REPLACE statement was also initially added to MySQL and is now also available in SQLite.

Today the SQL standard is very complex and there is no widely recognized authority that checks for compliance. This is the cause of much debate over the viability of the standard (Gorman 2001). It has also lead to a dramatic increase in size of the standard. ANSI-H2 member Joe Celko cynically quotes David McGoveran in this context: "A committee never met a feature it did not like" (Celko 2005*a*). As a result, the various implementations lag behind and do not always follow the specification because they implemented the feature with a different syntax before the standard added the given feature. Josh Berkus even goes so far to claim that "many vendors and users regard SQL-92 as 'the real SQL' and later versions as more of a library of exotic features to pick and choose from" (Appendix A.5.2). Joe Celko calls this the "Code Museum Effect" (Celko 2005*b*). The SQL standard also leaves the exact algorithms used in every implementation open to the given implementor. As a result, each RDBMS can behave much differently when pushed to its limits in terms of performance and features. It should be noted that the SQL standard is not concerned with the physical storage. As a result, the standard does not cover important aspects such as indexes, which are used to speed up most real world RDBMS installations, backup, monitoring and other administrative tools. The scope however remains large and even partial implementations tend to require large programming resources. For practical purposes most vendors ensure entry level SQL-92 compliance since this is the last version of the standard that was covered by conformance testing procedures. Later additions to the standard are only partially implement. A common standard API is however important in order to achieve portability of code from one vendor to another. Most programming languages do however provide additional layers, for example ODBC and JDBC, to help alleviate some of the portability issues.

Since the definition of the SQL-92 standard a lot of work has been directed at improving the performance and scalability of databases (SAP 2002). At the same time, open source databases have filled the feature gap which led to the creation of stripped down versions from all major proprietary vendors (Treat 2006*a*). Given the fact that there are several complete implementations of the SQL-92 standard available as open source or freeware, it seems indisputable that at least the lower end of the database

[2]Firebird choose to use different key words and position inside the SELECT statement.

market has been commoditized. This observation is shared by the answers in the interviews in the Appendix A, once again validating the theory that mature markets lean toward standardization and commoditization. Forrester research estimates that "80% of apps only require 30% of closed source database features" (Yuhanna 2006). In some specialized areas open source databases are even able to compete in higher end functionality as we will see in this and the following chapters.

There have also been three major trends in the database market in recent years. One is the appearance of so called object relational databases (ORD) in the 1990's and associated additions to the SQL:1999 standard. PostgreSQL and its earliest commercial off-shot Illustra, later acquired by Informix, were among the first to implement these concepts that, while inspired by, should not be confused with object orientation:

> "The object-relational aspect of PostgreSQL adds numerous enhancements to the straight relational data model. These include support for arrays (multiple values in a single column), inheritance (child-parent relationships between tables), and functions (programmatic methods invoked by SQL statements). For the advanced developer, PostgreSQL even supports extensibility of its data types and procedural languages.
>
> Due to this object-relational concept, tables are sometimes called classes, while rows and columns can be referred to as object-instances and object-attributes, respectively. [..] Other SQL data structures, such as indexes and views, can be referred to as database objects." (Worsley & Drake 2001)

However, these features are not must-have features (SAP 2002) while being sufficient to keep object-oriented databases (OODBMS) at bay for now (McClure 1997). What these additions have definitely done is increased the complexity of the supported syntax. Also, while PostgreSQL has been freely available for quite some time it only recently achieved enterprise level acceptance. Increased complexity and features mean an increase in required R&D resources which gives vendors reasons to keep or extend their license fees. This is a classic example of vendors overshooting market needs.

The second, more recent trend, is related to the success of database driven websites, "a market that isn't well suited to monolithic databases like Oracle and DB2" (Asay 2006*a*). As a matter of fact, today we see the second generation of the Internet emerging, which has been labeled Web 2.0[3]. The bulk of these web sites run on

[3]The term Web 2.0 was coined by Tim O'Reilly to describe the second generation of websites that foster information sharing and collaboration in new ways essentially displacing many applications that were previously run on the desktop locally (O'Reilly 2005)

large clusters of low cost servers (Taft 2006*b*). This also changes market requirements as suddenly the number of users a given database server has to handle is reduced as the workload is spread. So called shared nothing architectures are making inroads which require low connection overhead (Herrington 2003). At the same time, the need to improve the ability to handle XML and even less structured data inside SQL databases started being addressed. This is an area where incumbent vendors are more successful to date. For example, IBM's recent DB2 release highlights its ability to index and validate XML within the existing SQL interface (McCown 2006). It is still too early to tell how much real world market acceptance these feature will get. However, another important real world market trend has been the move from expensive large main frame computers to commodity hardware on which applications scale up instead of scaling out. These commodity clusters only work with low administrative complexity as many of the successful web sites were initially developed by small teams that lacked a dedicated DBA team. Also, these small teams often lack the knowhow to even make use of all of the SQL features introduced after the SQL-92 standard (Tomasi & Steppe 2006). This situation shows the key indicators of a market opportunity for a disruptive technology.

The third and final trend is in SQL database increasingly displacing custom persistent solutions, especially in the embedded market. In the past it was common to define custom formats and interfaces on top of flat files whenever the needs were either fairly simple or an SQL interface was deemed to be to inefficient. The "cost" for this was the need to develop and maintain this custom solution. Instead of being able to leverage common know how and tools, developers of such custom solutions were on their own. Interoperability with other products was also severely hampered. As prices in the database market have come down and features of embedded databases have expanded, custom solutions have become less feasible. This has even spawned the inclusion of entire SQL databases inside programming languages as we will see in the following chapter. Important to note here is that especially open source databases seem to be appealing in this area, since they do not add any licensing fees on top. For example, Sun mentions this as one of the reasons for their involvement in the Apache Derby project, which will be detailed in the section on Cloudscape (Section 5.5.2). The previously cited Forrester report therefore found that embedded applications are the most likely place open source databases are considered for (Yuhanna 2006).

## 4.3 Market Overview

While there has been considerable market consolidation that has left Oracle, IBM and Microsoft as the dominant market leaders with Sybase falling behind (Whiting 2003), especially open source database vendors hope to turn this around. There are vendors who provide purely proprietary solutions like Sybase. Microsoft is an example of a vendor who even has multiple proprietary solutions with SQL Server and Access. However, many vendors that provide proprietary solutions as well as open source solutions. Most of these databases were previously closed source. IBM is a good example of this with their DB2, Informix and Cloudscape lineup. The latter database was open sourced under the name Apache Derby after IBM acquired the database along with Informix (Shankland & LaMonica 2004). While Oracle has for a long time made it clear that open source databases are no competition to their high priced offerings[4], they have recently purchased InnoDB and Sleepycat and looked into getting MySQL AB on board as well (Shankland 2006).

On the other hand, we have open source companies who are trying to make a dent into the margins of these established companies. A number of these follow the dual-licensing scheme. The prime example here is MySQL AB, who also support MaxDB in partnership with SAP. MySQL AB has even bought closed source technologies and released them as open source afterward (Weiss 2003). In contrast there are also a number of vendors that solely provide support for open source licensed databases. There are also proprietary vendors that are using open source as their main code base like EnterpriseDB who compete with Oracle using the PostgreSQL code base (EnterpriseDB 2006*a*). With Borland we can also see a very peculiar case. Borland at one point open sourced their Interbase code base to spawn the Firebird project but later on decided to continue development independently again (Valderrama 2005). Another aspect about the database market is the breadth of different open licenses that were chosen: GPL (Ingres, MaxDB, MySQL), Apache (Derby), BSD (PostgreSQL) and product specific OSI approved licenses like the Sleepycat License (Berkeley DB) and the MPL based Initial Developer's Public License (Firebird). SQLite is even released as public domain software (SQLite 2004). This resulted in very different business models and development processes. Some vendors have chosen a dual-licensing model such as Sleepycat, MySQL AB and Ingres. On the other hand, IBPhoenix mainly provide related services while EnterpriseDB has created a proprietary solution around an open source base.

[4]Oracle was however one of the key open source adopter when it embraced Linux (Raymond 2001).

| DBMS Product | % of Enterprises using DBMS |
|---|---|
| SQL Server | 81% |
| Oracle | 77% |
| DB2 | 53% |
| Access | 28% |
| Sybase | 23% |
| IMS | 19% |
| Informix | 19% |
| MySQL | 17% |
| Teradata | 13% |
| Adabas | 9% |
| IDMS | 8% |
| PostgreSQL | 4% |
| Filemaker | 2% |
| Ingres | 2% |
| Progress | 2% |

Table 4.1: DBMS used in production (Yuhanna 2006)

Probably the best indicator of the increasing relevance of open source databases is the fact that Gartner recently changed their market share analysis methodology by including not only income from license fees, but all other forms of income as well such as support and training (Gartner 2006). Since companies like MySQL AB have an entirely different business model, it is hard to get proper estimates of the market relevance of MySQL by traditional metrics. Forrester took a different approach by looking simply at what kinda of databases organizations are using in production (Table 4.1). Market researchers Gartner and IDC do seem to agree that MySQL has a bright future ahead (Lombardi 2006). Zack Urlocker, vice president of marketing at MySQL AB even puts MySQL at number three due to its large user base (Urlocker 2006*d*). This still does not match the numbers from Gartner though, but they do expect these numbers to grow, with MySQL being the main driver for open source database adoptions (Urlocker 2006*b*). A market share analysis within the open source database world found that MySQL is clearly in the lead with 61%, Ingres and PostgreSQL follow with figures slightly above 10% and Berkeley DB being the only other database that is able to break the 5% mark (Figure 4.2).

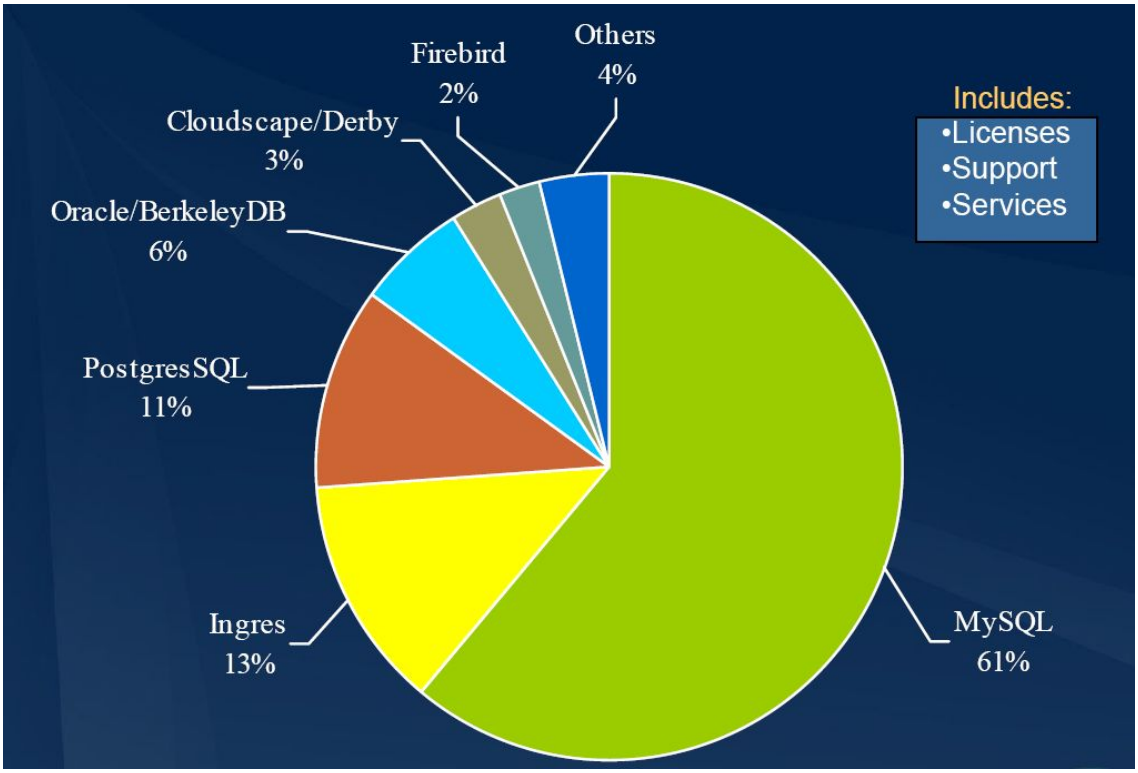


Figure 4.2: Open Source Database Market (Yuhanna 2006)

An interesting observation is the fact that all of the "Big Four" proprietary database vendors have all released free for development versions though artificially reduced in capabilities. Sybase started this initiative, SQL Server and Oracle followed and in early 2006 DB2 completed this development. One possible interpretation is that Sybase, as the weakest of the four, took this step in order to regain market share from its proprietary competition and that the others simply followed in order to preempt any shift. Maybe these new versions are more about open source though. IBM even directly approaches PostgreSQL users with a website to encourage them to leverage their skills on DB2 (Wasserman 2006). If judging by the answers of the interviewees in Appendix A, they see exactly this as the main reason. However, due to limitations, few feel very worried about these free as in cost databases: "They are a reaction to MySQL not PostgreSQL because the express versions cannot on any level compete with PostgreSQL" (Appendix A.5.4). They also note again that "free as in cost" of licensing is just one dimension and not necessarily the most important one. More

important is the freedom end users get, as well as the community that forms around the product as a result. We will revisit this topic in the section on Oracle's open source database activities (Section 5.3.2). Following is a non exhaustive overview in alphabetical order of major and emerging relational database products in the market. The selection is mainly based on relevance to the following chapters. Any important or controversial facts and statements are derived or referenced in other sections. The following chapter will expand many of the claims that are made in the following section.

### 4.3.1 Berkeley DB

Berkeley DB is an embedded database developed by Sleepycat Software, who were bought by Oracle in early 2006. Berkeley DB supports a large subset of the SQL standard and comes with a wide range of administrative tools. The database runs on most Unix as well as Windows operating systems and has bindings to most programming languages. Due to its nature as an embedded database it can be used as a general purpose data storage or via an SQL interface. Versions up to 2.0 were licensed under the BSD license, but today's version 3.0 follows the dual-licensing scheme with its own custom OSI proved open source license.

### 4.3.2 Cloudscape

This embedded SQL RDBMS is written in Java and is therefore platform independent. Originally it was developed by Cloudscape Inc., later it was acquired by IBM through their acquisition of Informix who in turn had bought Cloudscape Inc. earlier. IBM contributed the source code to the Apache Software Foundation in 2004 where it was released as Apache Derby under the Apache open source license. Sun's supported distribution of Apache Derby is called Java DB.

### 4.3.3 Firebird

Firebird is a fork of the open source version of Interbase which was released in 2000 under a variant of the Mozilla Public License. Since then it has been ported from C into C++ but remains very compatible to Interbase on the API level. It runs on most

Unix and Windows variants and features a very high level of standard compliance which even includes partial implementations of SQL-99 and SQL-2003. However, at this point it has been unable to establish a large world wide user base, but has been very successful in Brazil and Russia.

### 4.3.4 IBM DB2

The System R project was the starting point for DB2, which has continued to lead search and innovations in RDBMS. For example DB2 was the first database to add a cost based optimizer. It features very high SQL standards compliance and a large number of so-called enterprise features. In its early days it was solely used on mainframe computers but today it is available for most operating systems. DB2 is available in many different "editions" that cater to different requirements. In early 2006 IBM also released a stripped down version of DB2 which is free for development and distribution.

### 4.3.5 Informix

Informix was derived from the Ingres research project and was acquired and is still maintained by IBM since 2001. After once being second only to Oracle it is today no longer one of the key players in the SQL RDBMS market but is still well supported in programming languages due to its heritage. It offers a full range of advanced database features.

### 4.3.6 Ingres

Ingres was developed by researchers at the Berkeley University in the mid-1970s. Initially it was available in source code for a modest fee and served as the foundation for many successful other databases like Informix and Sybase. It was commercialized in the 1980's. After ownership of Ingres changed several times, it was finally acquired by Computer Associates who released it as open source after a few years. After then being spun out into its own company, called Ingres Corporation, in 2006 it is now available under the GPL and a closed source license, making it one of the most feature complete open source databases.

### 4.3.7 InnoDB

InnoDB developed by Innobase is mostly known for the MySQL storage engine by the same name. It quickly became the most popular transaction enabled storage engine in MySQL. It was acquired by Oracle in late 2005 but continues to be available for MySQL under both the GPL and a proprietary licenses deal between Oracle and MySQL AB.

### 4.3.8 Interbase

Interbase is owned and developed by Borland Software Corporation. The source code was made open source under a variant of the Mozilla Public License but Borland later decided to continue developing Interbase as a closed source product. Interbase introduced several innovative database features such as Multi Version Concurrency Control (MVCC) and Binary Large Objects (BLOBs). Aside from strong SQL compliance it also requires very little disk space. It runs on most operating systems and is supported by most programming languages.

### 4.3.9 MySQL

MySQL is developed and maintained by MySQL AB who license this database under the GPL as well as a proprietary license. It is considered to be the most popular open source database server, although it can also be run as an embedded database. It has mainly become popular as part of the LAMP software stack. It is available for most operating systems and programming languages. While its weakest points used to be standards compliance recent versions have improved the situation and have brought with them missing features like stored procedures and triggers which other RDBMS had long ago. One notable feature is the ability to easily add in alternative storage engines.

### 4.3.10 Microsoft SQL Server

Microsoft SQL Server was the result of a partnership with Sybase. The two parted however and Microsoft acquired the rights to the database for all versions of Win-

dows. SQL Server therefore only runs on Windows. The software is available under a proprietary license but there is a stripped down variant available which is free for development and distribution.

### 4.3.11 Oracle

The market leading database is Oracle. Its owned and developed by Oracle Corporation. It was the first commercial offering based on IBM's System R research project. It is available on multiple variations and also features a free for development and distribution edition. Oracle runs on all major operating systems from mobile devices to large main frames and is compatible with most programming languages.

### 4.3.12 PostgreSQL

This database emerged of the University of Berkeley as a follow up to the Ingres research project in order to bring in object relational features. Since its available under the liberal BSD license there have been several friendly forks some of which are even proprietary such as EnterpriseDB which adds several Oracle compatibility features. It is the most feature complete SQL RDBMS that was not previously closed source. It runs on most Unix operating system and since version 8 it also supports Windows natively. One of its key features is its extensibility.

### 4.3.13 SQLite

SQLite is a serverless flat file SQL database that is available as public domain software. It implements a fair share of the SQL-92 standard with the notable feature of being essentially data typeless. SQLite is relatively young, with version 1.0 only having been released in 2000. However, likely due to its liberal license SQLite has been quickly adopted by a larger number of open source projects and even proprietary vendors. Its available for a wide range of operating systems and programming languages.

### 4.3.14 Sybase

Sybase, also known as Adaptive Server Enterprise, is among the commercial spin-off's from the original Ingres code base. While it used to be one of the top SQL RDBMS products, it is falling behind today. This can be attributed to the unfortunate deal with Microsoft in which Sybase secured itself what it thought was the emerging large market for OS/2, which never appeared, in favor of giving Microsoft the full rights to the Windows market. The database is however still being developed and is also available for several Unix flavors unlike Microsoft SQL Server.

# Chapter 5

# Case Studies

## 5.1 Introduction

The following case studies serve to illustrate key aspects of open source and how it benefits companies and individuals who participate in the development of open source databases. The case studies are obviously not an exhaustive list of all possible case studies. For example, there is also a user community around MySQL even though the section on community only mentions SQLite and PostgreSQL. Furthermore, Ingres was also recently open sourced and is not mentioned in the relevant section. The sections themselves are grouped fairly loosely under titles to be easier to read and follow in a more logical order, but there is significant overlap between the sections. This is very apparent on the topic of support, which is relevant to all case studies and therefore there is no section labeled support. However, little effort is made to repeat important aspects of each of the sections in other sections. As a result, it may not be sufficient to read only specific sections and to skip others.

## 5.2 Community

### 5.2.1 PostgreSQL

Following the success of the Ingres research project, a new project was started at the University of California at Berkeley to implement an object-relational database in 1986. The name chosen, Postgres, illustrates this relationship. Due to its liberal

BSD license many commercial forks have been created since. Notable ones are Illustra as the first and more recently Bizgres MPP and EnterpriseDB. Parallel to all of this a community began to form around PostgreSQL in 1996 based on the original Postgres95 code base. This meant that all of the innovative and market leading object-relational features, some of which have not made it into other leading products even today, were available free of charge. The open source PostgreSQL project however continued to innovate with their unique MVCC implementation which sets itself apart from any alternative open source or proprietary implementation. They are also one of the few RDBMS to support transactions for Data Definition Language (DDL) statements. At the same time, the project kept pace with most of the key additions to the SQL standard such as ANSI joins, information schema. Standards compliance remains one of the key features. Further robustness was added by adding a Write-Ahead-Log, automatic LOB compression and page splitting and SSL connections (PostgreSQL 2006*c*). Through its object-relational architecture PostgreSQL was remarkably extensible, which led to the creation of a large array of third party extensions like full text indexing (GiST for PostgreSQL 2006) and GIS data handling (Refractions Research 2006).

The success of the PostgreSQL project, who sees itself as the second most popular open source database server (Appendix A.5.2), has attracted many commercial sponsors, adoptions and support offerings. Among the corporate sponsors are well known companies among which are such industry heavy-weights as Red Hat, Sun and Fujitsu (PostgreSQL 2006*d*). As mentioned before several companies have used the PostgreSQL code base to create their own proprietary solutions, which is explicitly allowed by the BSD license. Interestingly PostgreSQL offered many of the features users were missing in MySQL. While MySQL for a long time claimed a performance edge, albeit likely partially derived from these missing features, both projects have converged in terms of features and performance (Database Systems Group 2005). Of course due to fundamental differences in architecture[1] neither database will behave the same when pushed to its limits. Traditionally MySQL has however stressed ease of use, while PostgreSQL stressed standard compliance and architectural purity (Fosdick 2005). This is probably best illustrated by the fact that MySQL will soon drop the non "MAX" distributions (Arnö 2006*b*), which do no include all optional features, while PostgreSQL remains committed to providing a less feature rich, yet extensible, core distribution.

[1] MySQL is multi threaded contrary to PostgreSQL, which in turn has a much different MVCC implementation than provided by InnoDB, the only available stable MVCC-enabled MySQL storage engine at the time of writing.

| Name | License | Notes |
|---|---|---|
| Amalgamated Insight | proprietary | Fork of TelegraphCQ |
| Bizgres | BSD | PostgreSQL + BI features |
| Bizgres MPP | proprietary | PostgreSQL + BI features |
| EnterpriseDB | proprietary | PostgreSQL + Oracle compatibility |
| Great Bridge PostgreSQL | BSD | PostgreSQL re-distribution |
| Illustra | proprietary | Fork of Postgres |
| Mammoth | BSD | PostgreSQL + contrib modules |
| Netezza | proprietary | Appliance based on PostgreSQL SQL engine |
| NuSphere UltraSQL | proprietary | Native Win32 port of PostgreSQL |
| parACCEL | proprietary | PostgreSQL + BI features |
| Pervasive PostgreSQL | BSD | PostgreSQL re-distribution |
| PostgreSQL | BSD | Open source community project |
| PostgreSQL for Solaris 10 | BSD | PostgreSQL re-distribution |
| PowerGres | proprietary | Native Win32 port of PostgreSQL |
| PowerGres Plus | proprietary | PostgreSQL + custom storage engine |
| Red Hat Database | BSD | PostgreSQL re-distribution |
| TelegraphCQ | BSD | Data Stream oriented fork of PostgreSQL |

Table 5.1: Non-exhaustive overview of Postgres derived or re-branded databases

The original Postgres code base is the parent to more than a dozen open source and proprietary database products and distributions, most notably PostgreSQL (Table 5.1). To date none of the numerous PostgreSQL variants have displaced the core PostgreSQL distribution. Instead, they have increased credibility and provided resources for the PostgreSQL project, even though not all of them contribute back all or even some of their improvements. This ecosystem is a fundamental component of the project which is based on the fact that the BSD license makes it possible for companies to derive proprietary products from PostgreSQL. While this essentially precludes the opportunity to a dual-licensing business model it means that it is impossible for a single company to gain control over the entire source code. However, what companies have done is hire key developers in the community. Among the employers of the core developers behind PostgreSQL are well known companies Sun, Red Hat and EnterpriseDB (PostgreSQL 2006*b*). This both serves the purposes of being able to better serve their customers, assisting the project to add missing features and to

maintain credibility inside the community. The community itself has become fairly accustomed to this practice and recognizes this as a benefit, albeit one not free of risk (Tenney 2001).

These dangers became very apparent with the rise and fall of Great Bridge who were among to first to push PostgreSQL commercially on a pure support model. So instead of re-branding PostgreSQL they assembled a team consisting of several contributors including Tom Lane, Bruce Momjian and Jan Wieck from the core team (Great Bridge 2006). The core team itself had set the limit of three core team members at any single company (Tenney 2001). The company was very instrumental in putting the spotlight on PostgreSQL. It allocated resources toward improving PostgreSQL. One of the critical additions Great Bridge intended to bring to the project was replication (Tenney 2001). They released their research findings to the public but after only 16 months and $25 million in funding the company closed (Bruno 2001). It simply failed to get enough paying customers for their support offerings. Red Hat had offered them a partnership to support Red Hat's own Red Hat Database server that was also derived from PostgreSQL and that emerged after Great Bridge was started. However, the economics of the deal did not appeal to Great Bridge and by that time it was too late to find a buyer or investor (Shankland 2001). The scope of this fragile moment in the PostgreSQL history is best illustrated by comments from one of the core team members Josh Berkus:

> "In 2000, a start-up called Great Bridge hired most of the Core Team and a plurality of the major contributors. Then, for the same reasons as many others, this dot-com shut down in 2001. Suddenly, the people who wrote and maintained 80% of the PostgreSQL code were on the street looking for jobs.
>
> The PostgreSQL project faltered, then. If it weren't for the fact that our web host, Hub.org (and its sister company PostgreSQL Inc.) had refused to join in the Great Bridge venture, we might have died out. Or we might have become a captive project of Red Hat, only to dwindle a year later when RHDB was sidelined." (Berkus 2005)

However, the project did not end here. All of the core team members found new jobs. Bruce Momjian is now working for EnterpriseDB, Tom Lane moved to Red Hat despite the fact that Red Hat also discontinued their Red Hat Database product and Jan Wieck even completed a replication implementation[2] at his new employer

[2]There are many different approaches to replication `http://blogs.ittoolbox.com/database/`

Afilias (robertb@sraapowergres.com 2005). In the meantime with Pervasive another PostgreSQL company has failed to succeed (Farr 2006). However, this again can be attributed to the appearance of a strong competitor in the form of Sun (Loftus 2005). First Great Bridge closed shortly after Red Hat joined and second Pervasive announced their move out of the PostgreSQL business shortly after Sun joined. Both Great Bridge and Pervasive were focused on providing support and no further proprietary extensions. This empirical evidence is probably not sufficient to discount such a business model, but it does remain as a warning for smaller players. As illustrated in table 5.1, the ecosystem of commercial forks and contributors has still continued to grow since, despite these setbacks. As a result, the opportunities for PostgreSQL community members grow as well. Even if a single supporter falters, there is still the opportunity to not only continue open source development, but also proprietary integration. The latter would not necessarily be possible in the case of a GPL dual licensed company. Also, the main companies who failed were providing support rather than proprietary extensions which meant that customers only had to switch their support contracts not their installed servers. This further validates customer as well as open source developer investments into BSD licensed open source.

soup/archives/the-four-horsemen-of-replication-part-3-12140. Today there are also multiple distinct replication implementations available in PostgreSQL with Slony, Mammoth Replicator and PGPool.

### 5.2.2 SQLite

The serverless SQLite database is available under an even less restrictive public domain license:

> "The original author of SQLite has dedicated the code to the public domain. Anyone is free to copy, modify, publish, use, compile, sell, or distribute the original SQLite code, either in source code form or as a compiled binary, for any purpose, commercial or non-commercial, and by any means." (SQLite 2004)

While this makes SQLite technically open source, it does nonetheless illustrate how using an open approach to IP can affect software.

SQLite has been incredibly successful in the market, especially considering its young age. Following the initial release in 2000, came version 2 in late 2001 which added ACID transaction support and version 3 in 2004 which added UTF8 support. Today it is already being bundled with the Solaris, Mac OS X, Trolltech Qtopia, Symbian and Palm OS (ACCESS 2006) operating systems, it has made its way into popular software such as McAfee Anti-Virus and Firefox, has been bundled with the PHP language - thereby displacing the MySQL client libraries that were bundled until version 4 (Shankland 2004) - and has received corporate sponsorship from companies like AOL and Philips (Hipp 2006) (SQLite 2006*e*).

The peculiar detail is that SQLite has achieved this while having no marketing efforts to speak of. Running an Internet search for SQLite mostly provides links to projects using SQLite, developers blogging about using SQLite or other very technical articles. Direct searches on popular technology sites like ZDNet, eWeek and others provide essentially zero information dedicated entirely to SQLite. The only articles that can be found on these sites that mention SQLite are about products that have begun bundling or integrating SQLite (Taft 2006*a*) (Vaas 2005*a*). In 2006 Amazon lists just two books on SQLite specifically, though several books might mention SQLite due to it being bundled with a popular product. For example, most PHP5 books have at least short chapter or section on SQLite (Gutmans, Bakken & Rethans 2004).

As a result, most information for this section of the paper is derived from the SQLite website (SQLite 2006*d*), an email interview with Richard Hipp (Appendix A.6.1) and his replies on the Open Source Database Consortium mailing list (OSDB Consortium 2005). SQLite was developed by Richard Hipp in the classic "scratching your own itch" scenario. The initial version was released under the GPL since it was build on

top of the also GPL'ed GDBM library. With the rewrite away from GDBM he decided to move to a "more liberal license" but after looking at BSD style licenses he "could not see any advantage to these over just declaring SQLite to be public domain". Due to the increased success Richard Hipp's company Hwaci was able to fund Dan Kennedy as another developer for SQLite. Hipp estimates that he has written more than two thirds of the code, leaving close to one third to Kennedy and a meager 1% to "some dozen or more other contributors". Hipp notes that "the fact that people are required to sign a legal document in order to contribute to SQLite adds a barrier to entry that is sufficient to discourage many contributors, I suspect."

As mentioned already, SQLite has full ACID and UTF8 support. Triggers and views were added before MySQL, even though MySQL is almost twice as old. It should be noted however that now that MySQL has these features their implementation is more complete compared to SQLite. It is still missing support for some SQL-92 features, though some do not make sense for a manifest typed database and others, such as the lack of certain types of outer joins, can be worked around (SQLite 2006*c*). Probably the key missing feature is support for foreign key constraints. Due to the fact that SQLite is both serverless and manifest typed novice users are faced with a much lower barrier to entry. There is no server process to set up and the possible configuration options is small compared to other RDBMS (SQLite 2006*f*). SQLite benefits from ease of use with its type handling similar to how script languages like PHP ease the life for novice users by not requiring strict typing (Ascher 2004). While SQLite will run into bottlenecks when being faced with large tables and many client connections (SQLite 2006*a*) it does compare favorably in several real world scenarios even when compared to server driven open source alternatives (SQLite 2006*b*).

Hipp's company Hwaci makes its money by providing support, including annual maintenance subscriptions, custom programming and proprietary complements, like the Encryption Extension. The bulk of the custom changes is folded back into the public version. From this the company funds both Hipp's and Kennedy's development time on SQLite. As stated before there is no marketing, beside the SQLite website. In this way, SQLite employs a very common business model compared to traditional open source companies. Since the software is public domain, Hipp cannot make money from licensing his core product SQLite. The interesting fact is that Hipp "originally intended to release SQLite into the wild and to never make one dime from it". However, due to SQLite's success he could work full time on SQLite. Since SQLite only needs to afford the website hosting for its marketing activities the overhead for Hwaci is also incredibly low and there is no risk of failed investments for large marketing campaigns.

Given the liberal license terms, there is little reason for companies to be concerned when integrating SQLite into their products. For example, it is conceivable that Oracle could buy MySQL AB (Shankland 2006), stop selling proprietary licenses as well as cease development of the open source version. A similar scenario is not possible with SQLite. Due to the fact that Hwaci does 99% of the development on the database, it remains as the obvious choice when looking for support. Obviously the source code could be forked. However, even if the market for SQLite supports grows or the code base is forked it still seems likely that Hwaci will still be able to pull in a sufficient premium for controlling the source code of the well known and established SQLite "brand" to fund further development. The fact that SQLite source is available to the public at no cost also means that Hipp benefits from a large user community. As the number one benefit he lists "bug reporting" for issues he "would have never thought to look for".

As a result, Richard Hipp has secured the foreseeable future for his small company, that generates sufficient work so that he now also employs another developer, without having had to add any overhead and risk of a marketing and sales staff. At the same time, his work in SQLite has served as an "excellent resume" which has captured the attention of many potential employers should he one day find that SQLite is no longer generating enough income or he wants to move on to other challenges.

## 5.3 Dual-Licensing

### 5.3.1 MySQL AB

MySQL AB is one of the big winners in the open source world. So much so that Marten Mickos the CEO of MySQL AB felt confident enough to boldly brush off the acquisition interests of software and database heavyweight Oracle with the words: "We will be part of a larger company, but it will be called MySQL" (Shankland 2006). MySQL AB's main product is their relational database MySQL. Originally based in Sweden, MySQL AB employs over 300 people in various countries (Hyatt 2006). In its 11 year history it has become one of the most successful open source companies. Unlike other companies that have cashed-in on the rising of the Internet it has sustained its growth past the Dot Com Bust around the turn of the century. MySQL AB had a revenue of over $20M in 2004 based on a "100% growth for the past 3 years running" (Hudson 2006*b*). It is still sustaining that growth and so it is estimated that they will hit $80M by the end of 2006 (Asay 2006*b*): "The revenue growth is important, but even more so is that [MySQL AB] achieved this growth by spending less than 10% of total revenues on sales and marketing activities. By contrast, most public software companies spend 40-50% of total revenues on sales and marketing, and companies of MySQL's size generally spend 21.8% on these activities." (Asay 2006*f*, 108). MySQL AB claims that there are about 6 million installations worldwide with around 40,000 downloads per day (Hudson 2006*b*). Open source advocate Matt Asay goes so far as to put MySQL AB in the illustrious list of the first open source companies who have a shot at hitting the 1 billion mark in sales, right behind Red Hat (Asay 2006*d*).

MySQL is often shunned for its lack of support for more advanced features (Garret 2006) such as transactions, views and stored procedures most of which have only been slowly added over time. Nonetheless MySQL represents the "M" in the popular term "LAMP" (Linux, Apache, MySQL, and PHP/Perl/Python) that was coined by German journalist Michael Kunze in 1998 (Kunze 1998). Ironically even the popular Oracle FAQ uses MySQL (Urlocker 2006*c*). MySQL is released as open source, yet MySQL AB holds the full copyright over the source code. This is a critical component to MySQL AB's dual-licensing business model which is based on not only providing services to the open source users of MySQL but also selling proprietary licenses to companies unwilling to comply with the restrictions of the GPL open source license. As a result, there have been close to no code contributions from the open source community. So how does the MySQL AB business model work? Why are they considered

an open source poster example with so few outside code contributions? More importantly, how has MySQL AB been able to become such a successful business in this well established market?

One peculiar aspect of MySQL's success is that during its entire lifetime there have been technically "superior" open source alternatives. For example, MySQL only added transactions support in 2000 (MySQL AB 2006*a*). Support for maintaining referential integrity using Foreign Keys had to wait until InnoDB was first added in 2001 (MySQL AB 2006*b*). Features like subqueries were only implemented in version 4.1 which was marked as production ready in 2004 (MySQL AB 2006*c*). Support for views, triggers and stored procedures even had to wait until late 2005 to make it into a production ready release (MySQL AB 2006*d*). With this release MySQL also finally introduced the ability to validate data in a strict manner. The next version of MySQL will improve the situation some more, by adding support for partitioning, scheduling and XML. It will also improve the support for replication and clustering as well as logging. Check constraints are still missing, the subquery and stored procedure implementation is still lacking in performance, the prepared statement API not yet fully integrated. The most popular storage engine MyISAM is still neither transaction enabled nor does it support referential integrity. The list of additional other SQL features available in most RDBMS products yet missing in MySQL remains long. At the same time, there are still a fair number of features missing from so called enterprise ready databases such as more detailed monitoring and tuning of server internals, better support for LOBs in combination with XML data and materialized views. It should be clear by now that when looking at the feature set of traditional RDBMS, MySQL has a lot of trouble to keep up even today, let alone 5 or 10 years ago, when several products, including the open source database PostgreSQL, already featured transactions, subqueries, triggers, stored procedures, views (PostgreSQL 2006*a*).

At the same time, MySQL ensured it was able to sell proprietary licenses to its product. This meant that any code contributions would require that the copyright be signed over to MySQL AB. This was critical as even today MySQL AB is largely dependent on selling proprietary licenses (Hudson 2006*a*). As a result, the community provided code contributions are very small. As a matter of fact, when questioned on this at a LinuxTag panel in 2005 Michael Widenius, CEO of MySQL AB, could only remember one significant community contribution that was accepted. In effect, MySQL AB developed most of its code in house or acquired proprietary licenses from third party software companies. It makes its money from selling proprietary licenses, even though their product seems unable to match the feature lists of proprietary and

open source competitors. So what sets MySQL AB apart from any other proprietary software vendor?

MySQL proved to be a perfect match for the emerging web application market. While it was lacking many SQL features and was very lax about what data to accept, it was very easy to set up. It required little administrative work and knowledge to ensure good performance. As such, it was a great enabler for the large number of developers who were storming the Internet. These people often lacked the formal education that would make them capable of leveraging all the advanced SQL features of MySQL's competitors. MySQL has the "15 minute rule" that a new release of MySQL needs to meet: "The goal was and is to make it possible to install and try MySQL in 15 minutes" (Vaas 2005*b*). MySQL also features a very low connection overhead which has been one of the reasons of the LAMP stack success on the web (Dougherty 2001). Zack Urlocker from MySQL AB even claims that by their estimates "more than 90% of all Web 2.0 sites are powered by MySQL including the likes of media darlings like YouTube, FaceBook, Digg, Wikipedia, Cyworld, Flickr and most of the social networking sites" (Urlocker 2006*a*). MySQL was therefore good enough at solving the old database "problem", but it offered some significant new features which made an entirely different type of application possible. This is a classic example of a disruptive technology (Oram 2004).

MySQL also used a very elegant solution to overcome some of the more significant feature limitations. For example, a number of significant contributions did come from third parties. Their JDBC driver was actually one of many drivers developed in the community and they simply acquired the developer along with the full rights to the source code of the best one (Martens 2006*a*). Another contribution came in the form of the Berkeley DB and InnoDB storage engines which added much needed features like transactions and referential integrity. For each of these MySQL AB was able to make sure that their customers would also be able to get a proprietary license of these two storage engines. MySQL enabled this through supporting choosing different storage engines on a per table basis. MySQL has realized this as one of their key strengths. With MySQL 5.1 new storage engines can be integrated without having to recompile the core components even while the server is running (Oram 2004). MySQL will also certify these third party storage engines (Red Herring 2006*b*). This means MySQL is becoming sort of a technology platform for database developers who offer unique functionality. By turning their technology into a MySQL storage engine they quickly get access to a huge ecosystem of developers that can then transparently leverage this storage engine. MySQL is doing similar things for its full text index API (Gilfillan

2006). This will go a long way toward receiving more outside contributions to the MySQL database platform.

However, MySQL has benefited from the community in a very significant way even without code contributions: MySQL is being tested in real world scenarios by a huge community. These users provide bug reports, some of them even with analysis on the source code level (Hyatt 2006). Via the community MySQL has a close interaction with their user base. Since these are not paying customers they are bound by the GPL, which requires them to make the source of their applications available. As a result, MySQL AB has the opportunity to study the usage patterns in-depth in order to better determine the exact needs of its user base. This is a key challenge for any company to master. Also, it means that there are more applications for customers to use. Even if these applications do not require a commercial license, they will still create potential customers for support and training. Due to the popularity of MySQL, even if the bulk of these customers do not pay MySQL AB any money, the media press is also more interested in the MySQL database. This popularity breeds more popularity. As a result, MySQL AB needs to spend less on marketing. At the same time, customers can examine the product in much detail, or rely on third party analysis of the open source code. This means that MySQL AB sales people usually are faced with prospective customers that know the product well and simply need to be educated about the specific support services MySQL AB offers.

MySQL AB has recently started a program to solicit more community code contributions (MySQL AB 2006*f*). But this effort shows that working with the community is definitely a tricky challenge (Arnö 2006*a*). For example, when MySQL changed the license on its client libraries from the fairly lax LGPL to the stricter GPL license a huge uproar resulted. After many months of talks MySQL AB finally came up with a solution which soothed some of the tensions but has still left scars in community relations (Kerner 2004). Since then they have begun contacting community members prior to making announcements that could cause problems within the community[3]. With the creation of the My Forge website there is now an official place for documentation, code and concept contributions and collaboration (MySQL AB 2006*e*). MySQL AB also must direct a lot of resources to community relations. Kaj Arnö's community relations team consists of two to three people entirely dedicated to community communication around MySQL (Lentz 2006). Several MySQL employees are available on

[3]MySQL AB's community relations team has contacted me personally before announcing their MySQL Network support product and splitting their distribution into a free Community Edition and a closed Enterprise Edition

Internet relay chat (IRC) where they answer questions from end users essentially for free. That being said, the work on community relations also benefits MySQL AB by building the MySQL ecosystem.

Another aspect that MySQL AB has mastered is managing its work force (Hyatt 2006). The bulk of MySQL AB's workforce work from home, spread out over 25 countries. This enables MySQL AB to focus strictly on skill rather on having to make sure that each of their employees fits together in the office life. Employees can focus on their work, rather than having to compromise over their looks, work environment or any other aspects that distract them. This is especially important for top developers that are unable or unwilling to relocate. By employing open source development means which rely on IRC, peer review based on commit mails and meeting at conferences combined with weekly work journals, conference calls and a structure that ensures that the work done stays in line with the companies vision MySQL AB combines the best of both worlds.

For the long term MySQL will slowly chip away at the missing features. For example, MySQL will likely attain SAP certification sometime this year (Montalbano 2006). This will enable MySQL to move up the food chain to higher margin contracts. MySQL will need to make sure it does not alienate its user base in this process. For example, a key challenge will be staying committed to the "15 minute rule", though at the same time MySQL AB is trying to increase its support business (Appendix A.4.1). At first sight these two goals seem to contradict each other. MySQL AB could sell more support if they would make their product harder to use. Martin Mickos disagrees since in his opinion "popularity is worth more than the marginally improved fees you could get for a user-unfriendly product" (Mickos 2006). MySQL AB will try to circumvent some of the criticism of support business models that mainly revolves around the fact that providing more support usually requires a larger work force. For example, one of the components of the MySQL network is access to their knowledge base. Giving more users access to this data will not require a proportional increase in work force. Also, support contracts are turning more and more into an insurance business. The larger MySQL AB gets, the better they can spread the overhead of a 24/7 worldwide technical support infrastructure across all of its clients. Certification of third party extensions and sales partnerships also provide a new business opportunity.

MySQL AB will also be faced with a new type of competition. For example, due to the fact that the source code is open anyone can claim to know the source code just as well. Known performance experts, who previously have held jobs inside

MySQL AB, are using this as an opportunity to start their own MySQL consulting businesses(Zaitsev 2006). This could at least partial erode the current "monopoly" on enterprise support. Also, MySQL might eventually hit a wall if it does not pour more money into marketing. At this point it is simply not possible to say how far the reduced marketing and sales costs will get you, since there is simply no open source company that has broken the 1 billion dollar barrier. Finally, MySQL AB will need to prepare itself for the next disruptive technology. Further research on how MySQL AB will perform over the next years will be important to better assess the long term viability. However, MySQL's dual-licensing strategy has already proved to be incredibly successful in making MySQL a well known player in an already well established market. This is a solid foundation on which MySQL AB can build for the future. This has enabled them to gain access to the funds to they needed to acquire the technology that is now enabling their clustering solution (Weiss 2006) or the expertise of Jim Starkey when InnoDB was bought by Oracle (Darrow 2006).

### 5.3.2 Oracle Acquisitions

Oracle's acquisition of Innobase and Sleepycat sent a shock wave through the open source community (Vaas 2006*a*). Oracle's CEO Larry Ellison has the image of a fierce competitor that was quite outspoken against the offerings of open source databases vendors (Krill 2005). Most MySQL users were especially concerned about the future of InnoDB, the most popular ACID storage engine for MySQL, that is developed by Innobase. These concerns only grew when, with the acquisition of Sleepycat, Oracle suddenly also had control over the second ACID capable storage engine shipped with MySQL.

Innobase is a small company based in Finland. Its main product is the GPL'ed InnoDB storage engine that is bundled with MySQL. It adds a robust ACID transaction storage engine with support for row level locking, foreign key constraints and multi version concurrency control to the database. The later three are currently not supported by any production ready storage engine for MySQL. However, it relies on the MySQL infrastructure for SQL parsing, query planning and replication (Vaas 2006*c*). MySQL AB maintained a close partnership with Innobase that allowed them to also sell proprietary licenses for InnoDB. The two companies coordinated with each other on technical aspects. Innobase also created a proprietary hot backup tool for InnoDB. While InnoDB compares on equal terms with Oracle database offerings in several aspects (Dyck 2002), it is not able to compete against Oracle's high end feature set alone or when bundled with MySQL.

Sleepycat is a well established vendor in the embedded database market with its mere 25 employees (Lai 2006*a*). Actually, their Berkeley DB database is likely the most popular open source database, though less visible to end users due to its embedded nature. Aside from being used in many smaller devices, Berkeley DB is also bundled with essentially all available Linux distributions (Lai 2006*b*). Similar to MySQL AB and Innobase, Sleepycat employs a dual-licensing scheme for its products. However, they use a custom open source license called Sleepycat license (Sleepycat 2006). Interestingly earlier versions were available under the more liberal BSD license like many other software products that were originally developed at the Berkeley University (Ahmed 2001). The BSD license would have precluded a dual license scheme. Prior to acquiring Sleepycat, Oracle did not have an established base inside the embedded market, which is said to grow considerably in the next years (Brunelli 2006*c*).

Oracle of course is the leading vendor of relational database products. It was one of the key vendors to embrace Linux, and as a result open source, when there was still

little commercial interest (Raymond 2001, 183). While IBM has since taken the spotlight with its large investments into Linux and Apache, its open sourcing of Eclipse and pro open source marketing, Oracle has also allocated resources into Linux and ships products with the Apache web server. As a matter of fact, Oracle claims to employ more people working on Linux than the acknowledged leading Linux distribution vendor Red Hat (Aslett 2006). In the past two years Oracle has been aggressively pursuing acquisitions in the ERP and CRM market[4], where it competes in a battle over market dominance with the European software maker SAP. The rivalry with SAP is examined in some detail in the section on the open sourcing of SAP DB (Section 5.5.3). However, recently it also put it's eye on open source vendors, though the rumors over an immediate acquisition of JBoss and Zend did not turn into reality. JBoss was actually bought by long time Oracle partner Red Hat, which has led Oracle to consider to reevaluate their partnership (Waters 2006*a*). But through buying Innobase and more importantly Sleepycat, Oracle not only acquired open source products, it also acquired direct access to open source development and community expertise. It also became a subcontractor of MySQL AB, which will likely give them some internal insights into the MySQL community and its parent company. While Sleepycat is certainly interesting from a technology standpoint, since it puts Oracle into a new market, it is unclear how much Oracle stands to benefit from a technology standpoint from Innobase. Both certainly provide an opportunity to up-sell its other products and services once they have become integrated with Oracle's product lineup (Brunelli 2006*c*). The first release of Berkeley DB since the acquisition did however add major features such as MVCC, upgrades without downtime and a replication framework (Brunelli 2006*a*).

Even in spite of Oracle's important contributions to open source, the image in the community is not very positive. Larry Ellison, the CEO of Oracle, is known to be "outspoken" (AskMen.com 2006). Many people in the community tout MySQL against Oracle's database (Chamas 2005), in spite of the fact that even the CEO of MySQL AB says they can only compete with Oracle on the lower end (Shankland 2006). Recently, Ellison also went on record saying that "open source becomes successful when major industrial corporations invest heavily in that open source product" (Aslett 2006). One key point here is "major" as he discounts the ability of comparatively small companies such as Red Hat to take on such a role: "Red Hat didn't make Linux: IBM made Linux, Intel made Linux, Oracle made Linux" (Aslett 2006). The open source community would obviously claim that it made Linux. Ellison also predicted "an end to the

[4]ERP and CRM are applications that assist in managing documents, relations and other kinds of data collected inside an enterprise organization.

period of disruptive innovation from small technology companies" (Waters 2006*b*). The success of many of the recent Web 2.0 companies seems to contradict this prediction which Ellison seems unwilling to openly admit, but indirectly does by investing into these companies. As for the database market he feels it "is largely based on innovation, and those innovations are largely based on us" (Lacy 2006). However, he does consider himself a "believer in open source [where] open source can win" (Aslett 2006):

> "If an open source product gets good enough, we'll simply take it. Take [the web server software] Apache: once Apache got better than our own web server, we threw it away and took Apache. So the great thing about open source is nobody owns it - a company like Oracle is free to take it for nothing, include it in our products and charge for support, and that's what we'll do. So it is not disruptive at all - you have to find places to add value. Once open source gets good enough, competing with it would be insane." (Waters 2006*b*)

Oracle is also trying to transform its business from being focused on selling licenses, for which open source advocates frequently attack Oracle, into a subscription based model with selling software-as-a-service (Telegraph.co.uk 2006). With a similar approach IBM has turned around its business from being a hardware manufacturer to now being mostly a service company (Red Herring 2006*a*). Ellison finds this type of business very appealing: "Our margin dollars will increase at a higher rate with software-as-a-service. Plus there's no piracy, and no need to maintain old versions. There are huge advantages to the model" (Waters 2006*b*). More importantly, Oracle may find that its harder and harder to sell all of its innovation to users. Critics point out that most users do not upgrade because they simply do not need the new features or that the risk and cost of changing a running system to a newer version is not worth it (Knowledge@Wharton 2006). Therefore, software-as-a-service may be a long term solution to keep customers always running the latest and greatest Oracle versions rather than seeing them implement solutions in their own middleware. This would make them more independent from Oracle. Another appealing fact is that, at least in its current version, the GPL does not apply to software-as-a-service providers, since there is no distribution. As a matter of fact, many of the big open source users indirectly apply the software-as-a-service business model, like Yahoo, Google and Amazon, with incredible success. They all run services on top of customized open source software. To this end embracing open source could also be a means to learning and understanding how these companies are using this business model (Knowledge@Wharton 2006). Since many of these large companies, as well

as up and coming Web 2.0 companies, are using the MySQL-InnoDB tandem, Oracle will now have direct customer relationships with many of these software-as-a-service companies.

The timing of Oracle's open source acquisitions also seems oddly coincidental given the fact that MySQL 5.0 just went stable which added the bulk of the missing enterprise features to MySQL (Gilfillan 2005). The CEO of Sleepycat speculated that the Innobase acquisition was an attempt to "disrupt MySQL". Only shortly afterward he became vice president of embedded technologies at Oracle after his company was also bought up (Brunelli 2005). Now in a one two punch MySQL looses its most popular transaction capable storage engine, just to loose the only other a few months later to the same company. While MySQL's price is likely peanuts to Oracle, it might have been an attempt to weaken any opposition to the offer to acquire MySQL AB shortly afterward. Mickos, the CEO of MySQL AB, however declined the offer (Shankland 2006). This prompted Ellison to quickly point out the relevant insignificance of MySQL AB (Williams 2006).

Earlier that year Oracle none the less did make a free for deployment stripped down version of their database available, which many attribute to the growing success of MySQL and other open source databases (Clarke 2006). If such free versions are actually a proper answer to open source is questionable, since it seems unlikely that they will generate the same kind of community base (Asay 2006*e*). MySQL AB's reply seems to validate once again that you cannot buy open source (LaMonica 2006*a*). For example, MySQL made it clear that it would certainly continue fixing bugs in InnoDB, regardless of the outcome of license discussion with Oracle, which would solve some concerns by open source users (Vaas 2006*a*). A community "call to arms" brought to light that there was already an alternative storage engine in the works by a member of the community called PBXT (Bradford 2006). MySQL AB itself acquired Netfrastructure and along with it the services of database veteran Jim Starkey, who among many other things masterminded Interbase (Vaas 2006*b*). It also got one of their neighbors[5] SolidDB to work on putting their technology inside a storage engine (Lavigne 2006). Finally, Oracle and MySQL AB agreed to renew the license deal (LaMonica 2006*b*). So within less than a year MySQL not only renewed the license deal for InnoDB, it also compensated the loss of the fairly unpopular Berkeley DB storage engine with three promising alternatives, one of which they own entirely.

Another possible dimension to these acquisitions could be related to the Oracle's rivalry in the ERP and CRM market. As stated before, the MySQL database is close to

[5]SolidDB and some of MySQL AB's staff actually are located in the same office building in Finland.

becoming SAP certified (Montalbano 2006). Stalling MySQL AB in its efforts to bring a viable alternative choice for running SAP's application stack at a much lower license cost level could be one of the aims of Oracle. While MySQL is still far away of having all the advanced features in place with the same level of trust as Oracle has build up, its clear that MySQL AB is in a good position due to its close partnership with SAP. At the lower end of the ERP and CRM market Oracle competes with the likes of Microsoft[6]. The two companies also compete on the database front (Brunelli 2006*b*). Open source has a much better chance of taking away business from Microsoft at the low end than from Oracle at the high end. So Oracle might be trying to to cut into Microsoft's business using open source (Garry 2005).

At any rate, these acquisitions clearly show that dual license company owners can be quite interesting acquisition targets. This is an additional option they hold in their hands in order to make money for them and their investors. At the same time, they can count on the community to step up when there is trouble in the market. Potentially this community support could also be useful when trying to fend off an unwanted take over bid. While the proprietary customers of acquired dual license companies may find themselves in a similar situation as those of pure proprietary companies, they at least have the option of evaluating moving to the pure open source alternative. They can also hope to be able to rely on the community to maintain the source code should the new owners decide to discontinue the given product. Finally, if all else fails they can hire developers to take on maintenance themselves.

[6]Larry Ellison has also been very outspoken against Microsoft co-founder Bill Gates who he often refers to as "convicted monopolist" (Gilbert 2004).

## 5.4 Proprietary-Extension

### 5.4.1 PostgreSQL

PostgreSQL not only serves as a good example of how individuals can benefit from their involvement in open source (Section 5.2.1), it also illustrates how commercial companies can benefit from participating in open source products that employ a non-propagating license like the BSD license. Similar to Ingres, PostgreSQL has spawned a lively ecosystem of commercial vendors. While Ingres itself is open source today (Vaas 2004), all of its famous children remain closed source, with no ties to the original project. By contrast, all of the major spin-offs from the PostgreSQL code base remain major backers of the original project by employing key developers, sponsoring PostgreSQL related events or giving exposure in the press.

There are many companies such as Illustra, Pervasive, Red Hat, Sun, Fujitsu, SRA and several others that have or still participate in the PostgreSQL community. However, this section will focus on Command Prompt, Greenplum and EnterpriseDB. Command Prompt is the oldest of the three as it was founded back in 1997. They are a pure PostgreSQL focused company that donates resources toward the PostgreSQL project, provides an open source value added bundle of PostgreSQL with various optional tools called Mammoth PostgreSQL and sells a proprietary extension to this distribution for replication. Aside from that, they provide support for PostgreSQL which is by far their main source of income, although licensing does add up to 15% of their revenue (Appendix A.5.4). Therefore, Command Prompt is primarily a support and only secondarily a proprietary-extension company.

Greenplum followed in 2000 after having evaluated other open source database options to base their business intelligence technology on (Appendix A.5.3). The company employs 3 developers that solely work on the open source version, with a team of 12 working on building a highly specialized business intelligence closed source solution on top of PostgreSQL. As such, Greenplum's proprietary solution Bizgres MPP therefore indirectly benefits from the many developers from the community, even those employed by other commercial vendors in the PostgreSQL ecosystem. In order to more quickly move new features to both PostgreSQL and Bizgres MPP, the company also provides their own open source distribution of PostgreSQL called Bizgres. Contrary to Command Prompt, Greenplum generates the bulk of their income from licensing and support contracts for their closed source product. Interesting to note is that Greenplum estimates that 65% of their resources are directed at R&D, a

number that is around three times as large as the percentage numbers for Microsoft and Oracle mentioned earlier (Section 3.7).

In 2004 EnterpriseDB was founded. Their first product announcement in 2005 (EnterpriseDB 2006*b*). EnterpriseDB's business model essentially boils down to taking Oracle head on by providing a compatibility layer on top of PostgreSQL. Sony Online Entertainment migrated away from Oracle claiming that "80-90% of its applications run without modification", saving them around 80% of their database licensing fees (Bloor 2006). They have also closed a deal with Sun to provide support for the PostgreSQL distribution for Solaris (EnterpriseDB 2006*c*). With Bruce Momjian they employ a member of the core development team aside from several other PostgreSQL developers. They have also sponsored PostgreSQL events and funded a financial support program for PostgreSQL (EnterpriseDB 2006*d*). The idea of emulating Oracle is not novel as there are at least three other such products that predate EnterpriseDB with SAP DB, Ingres and Fyracle (Ruizendaal 2005). While Fyracle even claims the edge in compatibility it was unable to gain even close to the same popularity [7].

All three companies realize that they have to bear the additional cost of having to maintain their closed code additions, which may break over time due to changes in the main PostgreSQL source code. Therefore, it makes business sense for them to only keep the key differentiating pieces closed source while making sure all non-differentiating changes are applied to the open source distribution. This makes it clear that maintaining good relations with the open source community is critical to the success of their business model. However, at times they may find that their needs are too specialized for the needs of the general open source distribution. This is probably one of the reasons why Bizgres and Mammoth were founded. This way non-differentiating changes can also leverage open source testing and feedback, even if they do not get added into the main PostgreSQL distribution.

The community in turn recognizes the benefit these commercial companies bring to the project. They employ key developers, provide resources and money for conferences and increase media awareness (McConnachie 2005). However, maintaining this perception takes a lot of work. When asked about the biggest challenge in working with the community Joshua Drake from Command Prompt notes: "I think the biggest challenge is fighting the perception that companies are evil" (Appendix A.5.4). The community can also be quite demanding about where companies should invest their venture capital money at times (Treat 2006*b*). That being said, many people in the

[7] For example, a search on Google reveals only about 20.500 pages for Fyracle versus 1.630.000 for EnterpriseDB even though Fyracle has had a head start of over a year.

PostgreSQL community feel very strongly about the benefits of the BSD license, especially versus the GPL licensed used by MySQL: "The way MySQL AB uses the GPL gives it a commercial monopoly on the code. That's not freedom in my book" (Smith 2006). Within the PostgreSQL community the spirit seems to be that proprietary customers using PostgreSQL based products are getting a sound base, at a more reasonable cost. However, they need to be aware that they face most of the problems other proprietary customers face.

The companies writing these proprietary extensions will need to play nicely with the community in order to leverage their development model. When ever they decide to keep code closed source they will have to keep up with the open source version. Both sides realize this give and take relationship. This provides a sound and reliable basis for commercial companies to make further investments into the PostgreSQL ecosystem. The proprietary-extension business model therefore is very different from the dual-licensing model: Where dual-licensing mainly uses open source as a distribution method, proprietary-extension uses open source as a production method for the non-differentiating parts of the final product. Of course there is overlap in both models, like the need to work cooperatively with the community. The key factor is chosing either a non-propagating or strong propagating software license.

## 5.5 Open Sourcing

### 5.5.1 Interbase

Interbase was the first major proprietary RDBMS that was open sourced. According to Claudio Valderrama (Valderrama 2005), following the resignation of several key developers from its ISC subsidiary, Borland, called Inprise at the time, announced it would open source Interbase 6 in early 2000. It took another half year before that promise became a reality on July 25th. Legal issues are among the reasons for the delay. The source code was made available under a slightly modified Mozilla license though documentation was not included. Moreover, no external code was accepted and as a result only 6 days later the Firebird projects forks the code. With Ann Harrison and Paul Beach two key developers of Interbase form IBPhoenix which today is the leading commercial supporter of Firebird. In spring 2001, after a change in management at Borland, development of the open source version of Interbase is halted and moved to closed source development once again. Only a year later the first stable version of Firebird is released.

Since then both Firebird and Interbase continue to evolve with both being particularly popular in the Delphi community based on Borland's programming language products (Cantu 2006*b*). The products mostly remain compatible even though the Firebird development team decided to port the code from C to C++ for better maintainability (Beach 2004). This effort was largely completed with the release of Firebird 1.5 in early 2004. Aside from its tight integration into Delphi, Firebird and Interbase are able to draw users based on its broad support of the SQL standard and its ability to scale from small embedded to large setups. Firebird's user base seems surprisingly strong in Brazil and Russia (Cantu & Cisar 2005), although the numbers are likely slanted toward Brazil due to the fact that the survey was conducted by a Brazilian. Another interesting observation from this survey is the clear dominance of Firebird 1.5 over 1.0 usage, which indicates just how quickly open source users adopt new versions. Furthermore, Windows seems to be the dominant operating system platform. Even though Evans Data Corporation puts Firebird close to MySQL in terms of the number of open source database deployments (Marson 2005), it still does not get close to the same amount of press as MySQL and PostgreSQL. The cited report also only surveyed 400 developers. The numbers certainly do not match with other studies which put Firebird at 2% of the open source database market (Yuhanna 2006). But even that numbers makes it clear that Firebird is a notable open source database.

One of the surprising news to come out of the open sourced Interbase code was the discovery of a back door that was added in 1994. It would enable attackers to manipulate databases using the user name "politically" and password "correct". Firebird developer Frank Schlottman-Godde discovered the back door on December 18th 2000. It took 5 months since the open sourcing of the code base to uncover this vulnerability that has been in the code for 6 years already. The back door was implemented to solve issues created by adding a database internal authentication scheme: "As long as nobody knows about this back door, it works. It's still secure," says Schneier. "But as soon as somebody finds out about it, everybody is immediately and irrevocably insecure." (Poulsen 2001) This is known as security through obscurity, as it relies on facts being unknown, rather than actually being secure. Given the relative ease of exploiting this issue and the fact that developers inside Borland had access to this code even before the open sourcing, it seems questionable how obscurity could be considered to be a valid approach. It is also surprising that Borland did not take steps to improve security before releasing the source code as it was clear that it was only a question of time until someone would discover this back door. This would indicate that decision makers inside Borland were not aware of it and probably only few developers knew about it.

The important realization from this is that no secrets in the source will remain secret once the code is out in the open. It is also not possible to undo open sourcing of a closed source product. Borland's strategic mistake seems to be have been that they attempted to control the source code but they chose a license unfit to this purpose. The Mozilla license made it possible for others to integrate Firebird even in proprietary products. As such, there was no stop gap to discourage companies to pick up the source code and run with it. This also effectively inoculates Firebird from ever being dominated by a single entity (Cantu 2006*a*). More importantly, Borland was unable to keep their key developers happy inside their company. So given the open source code, these developers decided to take their future into their own hands by founding IBPhoenix. This company even managed to eventually subcontract development of a 64bit and SMP capable future version of Firebird to Jim Starkey, who started Interbase initially (IBPhoenix 2003). A testament to the commercial viability of IBPhoenix.

The relationship between Interbase and Firebird remains tense. Holger Klemt's view of the situation is that "Borland ignores the existence of their open source product since the time right after opening the source. It seems that they still think that a database server or client license can be sold in the future" (Appendix A.2.1). Marco Cantù, well known Delphi expert, even goes so far as to say that there have been "per-

sonal fights in the past" (Cantù 2006). However, in the same blog post he expresses the hope that with the creation of yet another spin-off from Borland, currently codenamed DevCo, there might be a change of relation for the better. Even with the recent shakeup after Jim Starkey joined MySQL fulltime and Ann Harrison halftime (Cantu 2006*c*), Firebird still seems to have a bright future. Jim Starkey finished the work on the Vulcan project for IBPhoenix. The eminent release of Firebird 2.0 and the Vulcan code, which is to become Firebird 3.0 eventually, should help increase the momentum (Firebird 2005). This is good news to the commercial companies inside the Firebird ecosystem like IBPhoenix and HK-Software.

### 5.5.2 Cloudscape

The open sourcing of Cloudscape is a straight forward success story. Cloudscape is an embedded RDBMS implemented entirely in Java. The database started its life in 1996 at Cloudscape Inc., but ended up at Informix in 1999. With IBM's purchase of Informix the database again switched owners in 2001. In 2004 IBM decided to open source the source code of Cloudscape. However, instead of creating their own project, IBM decided to cooperate with the Apache Foundation. IBM already had close ties to the Apache project through their involvement with other subprojects, most notably the Apache Web Server. In 2005 the Cloudscape source code made its way out of the incubator phase under the name Apache Derby (Appendix A.1.1).

Handing the code over to the Apache Foundation proved to be a smart move by IBM. It gave the project an established open infrastructure to leverage. This applies to the project web site and source code management, as well as legal aspects like the Apache license and contributor agreement (Apache DB Project 2006). Also, the project benefited from the Apache brand name for better visibility and recognition. While IBM still employs key developers, such as the initial founder of Cloudscape Inc. Daniel Debrunner, there are a number of non IBM developers working on the source code (Debrunner 2006). At the same time, Cloudscape is no competition for IBM's other database products Informix and DB2. Instead, Apache Derby has the potential for applications, that have only simple persistence needs, to even consider using an SQL database instead of a custom solution. These applications can then scale up using DB2, since IBM has made several changes to Cloudscape to ease such migrations (Appendix A.1.1).

Most notably Sun also joined the project in 2005. Since Apache's license forbids distribution of modified versions of Apache Derby under the name "Derby", Sun decided to call their product Java DB. However, Sun has the full intention of making all their code changes part of the official Apache Derby release, though they may at times decide to add third party modules. What is even more significant though is that Java DB is slated to be included with the Java 1.6 SDK (Orsini 2006) similar to the SQLite integration in PHP. In a way this is somewhat surprising since OpenOffice, which is also heavily funded by Sun, had already decided to bundle another Java database called HSQLDB with OpenOffice 2.0 (hsqldb Development Group 2006).

From the point of view of IBM this must be a great success. By open sourcing Cloudscape they have been able to bring a significant code contribution into the Java language. IBM benefits from the fact that they have some level of control over what hap-

pens in Apache Derby, although whatever influence they exert needs to pass through the open process of the Apache Derby project. They also benefit from a marketing perspective.

### 5.5.3 SAP DB

Most of the incumbent RDBMS were derived from either Ingres or System R. SAP DB is one of the few exceptions to this rule as it was originally developed as a research project at the TU Berlin. After the project started in 1977 it became a commercial product at Nixdorf, Siemens Nixdorf, Software AG and finally SAP. As a result, its known under various names. Today its maintained under the name MaxDB by MySQL AB following the open sourcing of SAP DB in 2000, though SAP is still much involved in the development (SAP 2002). The database features essentially everything one would expect from an enterprise database with the notable exception of real time replication support which is not even planned to be added at this point (Dittmar 2002). This means that in order to scale to higher loads installations of MaxDB rely on having separate database servers per application and using large SMP machines. It can also claim unique features in the area of automatic system maintenance which make this database particularly suitable for so called 24/7 shops. This feature is a key vision for all future developments (SAP 2002).

SAP itself states several reasons for why they open sourced their database. One of the obvious reasons is that SAP wanted to exit the database market (Darrow 2003). They are focused on their application business that sees databases as a necessary, yet non-differentiating, compliment of their core business (SAP 2002). Any steps to lower the costs of their applications are very welcome, similar to how Oracle benefits from running on their database on top of Linux (Asay 2006*c*). Using community feedback, especially from users outside of the SAP ecosystem, to lower the costs of product improvements is also a very appealing proposition (SAP 2002). However, these efforts failed to generate much success. By 2000 the market for open source databases was already quite crowded. Other problems like the non standard build system and the unwieldy source code, a mix of pascal and C code (Anonymous 2002), and the somewhat hard to navigate documentation (Pavlicek 2004) did not help either. While SAP employees can partially refute these observations, convincing the community at large is an entirely different matter (Dittmar 2002). This in turn meant that there was little reason for book publishers to create books on MaxDB ("Fished" 2002). The result is the classic chicken-egg-problem. In a move to overcome some of the challenges in building a community, SAP joined forces with MySQL AB in 2003 (Pavlicek 2004). As part of the deal the database was released under the GPL with the new name MaxDB. MySQL AB got the rights to sell proprietary licenses and use the source code to add new features to its own MySQL database (Songini 2003).

MySQL AB in return gives MaxDB much needed credibility in the community. Their expertise in open source community relations and interaction also help to overcome some of their issues. For example, the MaxDB documentation was brought under the same structure as used for MySQL (Pavlicek 2004). A proxy was released to make it possible to use all of the MySQL applications with little changes with MaxDB (MySQL AB 2003). At the same time, SAP still keeps close to 100 developers working on the source code (SAP 2002). Also, SAP and MySQL AB would join forces to develop a new database (Hoffmeister 2003). It is not entirely clear if this is still being pursued. The existing MySQL 5 database is getting SAP certified already (Montalbano 2006). Either way, this deal gives MySQL AB much more credibility in the business world, where SAP is a recognized leader (Siteforum 2003).

SAP has made it clear that it wants to help in commoditizing the database market (Salkever 2003). In their opinion feature-wise there is little relevant differentiation in the market and only performance has so far separated proprietary from open source solutions (SAP 2002). By releasing SAP DB as open source they are hoping to close this gap. For a very long time SAP has based its development on Oracle databases who's licensing fees have become sort of a "tax" that simply increases the price of SAP based solutions (Walli 2006). Oracle is being used in over 60% of all SAP installations (Darrow 2003). In the future SAP plans to develop all of its applications for SAP DB first (SAP 2002). Obviously cutting out the license fees for a database means that SAP solutions would automatically come down in price. The relative cost of their 100 developers is insignificant to their sales volume of over 7 billion (Salkever 2003) and the high licenses fees for Oracle's database (Cone 2003).

Oracle, and to some extend Microsoft, are attacking SAP on their home turf, the enterprise ERP and CRM market (Sanders 2006). Oracle in particular has been cross-subsidizing their numerous acquisitions in this field, frequently outbidding SAP, using their database as a cash cow (Martens 2006*b*). Oracle and SAP are fairly open about this battle. Both companies have specific license deals to encourage users to migrate to them from the other side (Gilbert 2005). As a result, if SAP is successful in commoditizing the database market it would strike a major blow to its most fierce competitor Oracle who would no longer have the means to buy itself more customers. There have been other similar efforts. For example, Sun released OpenOffice as open source which could likely be interpreted as an attack on one of Microsoft's cash cow products Microsoft Office (Loftus 2004). While OpenOffice has not managed to displace Microsoft Office, IBM has been more successful with the open sourcing of Eclipse. IBM realized it would not be able to profit from selling Eclipse. Therefore, they open sourced it in an effort to reduce its development costs and at the same

time, scorched the market for Java development IDE's (Woods & Guliani 2005, 11). However, SAP's CEO Shai Agassi claims that they are "not contributing to commoditization to spite Oracle" and that instead they do it "because it's the most logical way to reduce the costs of ownership" (Salkever 2003).

To date MaxDB is still only used by less than 5% of SAP customers (Appendix A.3.1). In that sense open sourcing of SAP DB has not yet become a success. These numbers might change though as new products by SAP will be build with MaxDB in mind. Also enterprise customers are notoriously slow in making changes even if they are continuously looking for new ways to reduce their costs. SAP has not really forgone any significant income anyways, so it has not really lost much if this change will not take place. However, what it has done is gained some more credibility in the open source community and also lend credibility to MySQL AB in the business world. This might help SAP in the long run as MySQL AB looks to be one of the major players that could help in database market commoditization.

# Chapter 6

# Conclusion

## 6.1 Key Findings

Probably the most surprising finding is that open source databases and proprietary offerings do not seem to contradict each other. They can even co-exist within a single company like in the case of IBM and Command Prompt. Open source business models also do not invalidate basic economic principles. Based on this paper it is not possible to prove if open source databases will eventually displace all proprietary software. There are no significant indicators for this. However, prices will and have already being going down. The best example for this are the free for deployment versions of popular proprietary vendors. The key realization is that more and more of the database market will become commoditized which will erode the lock-in proprietary vendors use to be able to charge a premium.

Open source companies rely on reduced transaction costs as well as optimization and innovation, especially in niche markets. This makes them able to reduce their costs instead of relying on customer lock-in to keep prices high. MySQL is very successful in the Web 2.0 market. Several open source solutions are pushing into the embedded database market. These companies leverage the community to lower their costs of development. Dual-licensing companies turn to the community for testing, while proprietary-extension companies cooperate with the community for development of non-differentiating parts of their proprietary products. As a result, both types of companies can build a production cost edge over traditional software manufacturers. Beyond licensing they also use open source to grow the market for their complimentary offerings such as customizations, support and training.

The RDBMS market is particularly suited for open source vendors. For one, the market is mature, which is an important facilitator for market commoditization. Furthermore, the database market is a classic example of the innovators dilemma. Database vendors, in their hope to sell upgrades, continued to add more and more features that fewer and fewer customers could actually use. At the same time, SQL databases are being used increasingly in entirely new scenarios, most notably the Internet and embedded applications. This was a perfect opportunity for new vendors to offer "good enough" solutions which are highly optimized for these two quickly growing niche markets.

Another interesting trend is the open sourcing of previously proprietary databases which is one way for incumbent vendors to react to this emerging open source trend. At the same time, big database vendors like Oracle are investing in open source databases. The reasons and goals for taking this step differ greatly, though a definitive answer for these steps is hard to find since companies obviously do not make all their strategic decisions public. Reasonable speculation would indicate however that among the reasons for open sourcing or acquiring open source one can find the following:

- Create or expand a market for complimentary products and services (SAP DB, Cloudscape, Interbase)
- Increase pressure or even destroy the market for competitors (SAP DB)
- Learn about open source (Sleepycat, SAP DB)
- Leverage community for reduced development costs (SAP DB, Cloudscape)
- Get feedback from the community (SAP DB, Cloudscape, Interbase)

The important lesson one can learn from the Interbase example is that it is very important to choose a license that is in line with the goals of open sourcing. Furthermore, the failures of some of the PostgreSQL support offerings from mostly pure play open source vendors, that neither use dual license or proprietary-extension, indicates that this is a somewhat fragile business model. However, companies like Command Prompt, Hwaci and IBPhoenix show that as long as expensive marketing efforts are evaded, even a business model mostly based on support and custom programming can bring long term success.

The community stands to gain in several ways as well. Most importantly commercial companies bring money into the ecosystems of each of the open source databases. This results in employment opportunities for core members. It also helps in funding

hosting, conferences and other community related activities. Commercial involvement also results in increased press coverage. Finally, commercial vendors provide the much needed enterprise support offerings that customers require which in turn expands the ecosystem for the given open source product.

## 6.2 Future Research

During the work on this paper several areas for further research became apparent. First, there seems to be very little reliable data about market shares of open source databases. Forrester, Gartner and IDC seem to be working on improving this situation. Their methods and results are not freely available however. It will also be interesting to see how the free proprietary versions fare against open source offerings in the years to come. Also, will this trend of open sourcing proprietary solutions continue? In this context it would also be interesting to study what feature set customers actually rely on most in their chosen RDBMS and how much of these features are available in open source solutions. Furthermore, how will the standard continue to evolve given that open source databases increase their market share, yet have no voice in the SQL standard definition process? Also, what is the significance of programming language database API's like ODBC, JDBC, Hibernate and LINQ to overcome incompatibilities as Gavin Sherry noted in his email interview (Appendix A.5.1). Another interesting trend to watch is the bundling of entire SQL databases with programming languages. With reduced licensing fees, it would also be possible to bundle databases with software or even full application appliances instead of requiring custom data persistence solutions or leaving the choice of RDBMS to the end user. For example, the recently open sourced database Ingres is now made available as a database appliance called Icebreaker (Ingres 2006*b*). Finally, a study of the difference between open source and closed source communities could give some further insights. The quality and quantity of bug reports could be an example of this.

## 6.3 Closing Comments

Looking for answers in any particular field always seems to result in more questions being raised than answers being found. The previous section already lists a fair number of new questions which would each warrant an in-depth study with a scope

similar to this paper. What this paper has done however, is demonstrate the growing significance of open source in the database market. It explains how the specific situation in the database market was important for this development and the strategies used by companies to succeed in this market. These observations should help asses how open source can work in other markets. It also reminds us that even in engineering, market innovation can occur on the business model side and not only on the technology front. Finally, as open source is poised to expand its scope beyond software development, the importance of open source and the speed of development can only grow in the years to come. As as result, this paper was updated frequently all the way to the final weeks before publishing, to reflect the most recent developments in the market.

# Glossary

| | |
|---|---|
| ACID | A transaction should be Atomic, Consistent, Isolated and Durable |
| ANSI | American National Standards Institute |
| API | Application Programming Interface - interface that a computer or library to provides in order to allow interaction |
| ASP | Application Service Providing - providing computer services for customers over a network |
| BI | Business Intelligence - applications that assist in enterprise decision making. |
| Binary code | Statements in machine readable form usually to describe a computer document or program |
| BIND | Berkeley Internet Name Daemon - most commonly used DNS server on the Internet |
| BLOB | Binary Large Object - a binary file stored inside a database |
| BSD | Berkeley Software Distribution open source license created at the Berkley University |
| Clustering | Combining multiple computers into a single large computer |
| CRM | Customer Relationship Management |
| DBA | Database Administrator |
| DDL | Data Definition Language - computer language to define the structure for data |

| | |
|---|---|
| DNS | Domain Name System - translates domain names to IP addresses |
| ERP | Enterprise Resource Planning |
| FAQ | List of frequently asked questions |
| FLOSS | Free/Libre/Open-Source Software |
| Foreign Key | Fields that point to fields in another table as a means of maintaining referential integrity |
| Forrester | Market research and advisory company |
| FreeBSD | Unix-like free operating system |
| Gartner | Market research and advisory company |
| GIS | Geographic Information System |
| GNU | Recursive acronym for GNU's Not Unix |
| GPL | GNU Public License |
| IDC | International Data Corporation - Market research and advisory company |
| IDE | Integrated Development Environment |
| IHV | Independent Hardware Vendor |
| Index | Method to improve performance for record lookup in databases |
| IP | Intellectual Property |
| IRC | Internet Relay Chat |
| ISV | Independent Software Vendor |
| IT | Information Technology |
| JDBC | Standard API in Java for interacting with RDBMS |
| Join | Combining records from multiple tables in a single query |
| LAMP | Combination of Linux, Apache, PHP/Perl/Python and MySQL - a popular platform for web applications |

| | |
|---|---|
| LGPL | Lesser or Library GNU Public License |
| LINQ | Language Integrated Query - standard API in .Net for interacting with RDBMS |
| Linux | Unix-like free operating system |
| LOB | Large Object - large file stored inside a database |
| Locking | Method to manage concurrency in accessing data in databases |
| Logging | Practice to record data access |
| MVCC | Multiversion Concurrency Control - method in databases to prevent reading transactions causing locks |
| ODBC | Open Database Connectivity - standard API for interacting with RDBMS |
| OS | Operating System |
| OSI | Open Source Initiative |
| Partitioning | Logically separating data from databases or tables to improve manageability or performance |
| PC | Personal Computer |
| RDBMS | Relational Database Management System |
| Referential Integrity | Consistency between coupled tables |
| Replication | Using of redundant resources to improve performance and reliability |
| SAMBA | Open source implementation of Microsoft's networking protocol |
| Sendmail | Popular email server |
| SMP | Symmetric Multiprocessing - architecture to allow multiple identical CPU's in a single computer |
| Source code | Statements in human readable form that can be converted into a machine readable form |

| | |
|---|---|
| SQL | Structured English Query Language - standardized interface language implemented in most RDBMS |
| SSL | Secure Sockets Layer - cryptographic protocols to provide secure communications on the Internet |
| Stored Procedure | A subroutine physically stored inside a database |
| Subquery | A query embedded within another query |
| TCP/IP | Combination of the Transmission Control Protocol and the Internet Protocol used on the Internet |
| Transaction | Method to group multiple statements into one atomic unit in databases |
| Trigger | A piece of procedural code executed as a result of specific queries in a database |
| UNIX | Popular operating system that is especially popular on servers |
| UTF8 | 8-bit Unicode Transformation Format - used to represent western as well as asian and any other characters |
| View | A logical table in a database derived from a query |
| XML | Extensible Markup Language - general purpose markup language to facilitate data exchange in heterogeneous environments |

# Appendix A

# Email Interviews

The following interviews were conducted during the time frame from end of June 2006 until end of August 2006 via email by myself. Unfortunately, some of the email interviews were not returned, mainly due to time constraints and are not included as a result. All participants were emailed a set of questions that were partially individualized to highlight specific aspects of the projects they are involved in and to adapt to possible time constraints. They are listed here unchanged[1] categorized by the parent project the person in question is associated with. I also conducted an informal telephone interview with Mark Townsend (Database Product Manager at Oracle) and Christopher Jones (Linux Engineering Team at Oracle) on April 24th 2006 in preparation to this paper.

Following is a short summary of the answers. For the most part the projects lacked so called hard market data facts. They do not feel like they are competing with other open source databases all that much. At most they compete with proprietary vendors, though for the most part they are hoping to gain new users through growing the overall SQL market. However, most projects seem content with filling a niche. As a result, there are mixed feelings about cooperation with other open source databases. Some feel that due to different focuses there is little sense in cooperation, others are hopeful to increase cooperation in the future. Security seems to be one possible area for information sharing. However, no project mentioned the potential for code exchange. Modularization is not seen as something that is required for open source in particular, but instead is simply good development practice. The necessity for following standards is recognized by all. The fact that the official ANSI standard cannot be influenced is not seen as an issue. Most also agree that the database market is already being commoditized. The fact that proprietary vendors have released stripped

[1] Only obvious spelling mistakes were fixed for better readability.

down versions which are free for deployment is seen as a sign of the success of open source. At the same time all projects were able to name major commercial users. As to the greatest benefits of being open source most projects name bug reports and other feedback from end users.

# A.1 Apache Derby

## A.1.1 Rick Hillegas

Q: Could you briefly introduce yourself and your role inside the Java DB project?

A: My name is Rick Hillegas. I have spent around a dozen years working on database internals for a number of companies, including Ingres, Sybase, Cloudscape, and Informix. I joined Cloudscape early on during its startup phase and contributed significantly to its LUCID synchronization subsystem.

Today I work for Sun Microsystems as a Senior Staff Engineer dedicated to the Apache Derby project. I am a Derby committer and member of the Apache DB PMC.

Q: What are the primary features that set Java DB apart from other RDBMS products?

A: Java DB is Sun's supported distribution of Apache Derby, available at no cost under the Apache license. It is not a code fork and Sun's development work is contributed back to the Apache Derby project. (The Apache License specifies that Derby code with any modification to the current release cannot be called Derby.) The Java DB release (and IBM's Cloudscape) use the Apache Derby code base, bit for bit, but may also include value-add modules or the newest patches or updates that have not yet been made available in official Derby releases. Users who wish to use Java DB in production may purchase support from Sun. Sun bundles Java DB in a number of products, including NetBeans, Creator, Sun's Application and Portal Server, and Project Glassfish. Java DB will also be included in the next major release of Sun's JDK.

The following features distinguish Java DB, Cloudscape, and Derby from other RDBMS products:

1. Small footprint, lightweight, 100% pure Java database.

2. Platform independence.
3. Simple to install, easy to use.
4. Embeds invisibly and does not require any system administration.
5. Ideal for shrink-wrapped, data-rich applications like cell-phones and downloadable browser apps.
6. Nevertheless, also runs in a client-server configuration and supports departmental-scale applications.
7. Feature rich, standards compliant, fully transactional.
8. Tightly integrated with the Java language.

Q: Could you give a brief history overview of Cloudscape? What is the differences, if any, between Cloudscape, Apache Derby and Java DB?

A: Here are a couple key events in Cloudscape's history:

1. Founded in 1996 by ex-Sybase engineers. We built:
   a) a 100% pure Java database
   b) targeted at zero-admin, embeddable apps
   c) ideal for downloadable, data-rich tear-off databases
   d) complying with the ANSI SQL and JDBC standards
   e) tightly integrating Java ADTs and methods in the query language
   f) supporting Hub and spoke LUCID synchronization
2. Beginning in 1998, attempted an enterprise play–but failed.
3. Acquired by Informix in 1999. Repositioned as a lightweight satellite database synchronized against Informix Hubs.
4. Acquired by IBM in 2001. Features added and removed to become compatible with DB2.
5. Open-sourced as Apache Derby in 2004.
6. Sun joined project in 2005
7. Derby released from Apache incubator as official project in 8/2005

Q: Could you explain how the Java DB project works? How is the code developed? How many developers are part of the project and what are the main companies that employ them?

A: Derby development proceeds under Apache's rules:

1. Discussions occur in public on Apache mailing lists. Occasionally, this email community votes on significant proposals.
2. Many developers submit patches and features. Successful submissions are generally well documented and small enough that reviewers can understand the changes.
3. Other developers review submitted patches, suggesting improvements.
4. A small number of committers control write-access to the Apache code line. These committers check in changes which look good and which pass regression tests.
5. From time to time, the email community votes to produce a new, official release of Derby.

Scores of engineers actively contribute a to Derby. Sun and IBM have both donated teams to this effort.

Q: The Java DB project was the result of open sourcing of a previously closed source database. What sort of cooperation is there still with IBM?

A: IBM cooperates through the community presence of its employees. Cooperation occurs in public through the Apache mailing lists.

Q: How the involvement in the open source Java DB affected your professional career?

A: Open source has taught me a lot about community-based decision making and distributed development. It has introduced me to broad networks of talented engineers. It has impressed me with the high quality of open source code and the value of global, public discussion and review.

Q: What is Sun's business models around Java DB?

A: Many of Sun's products incorporate database technology. Derby, as an all-Java, full featured database available under the Apache license, met many of our products' needs nicely. Standardizing on Derby wherever possible provides Sun significant cost-savings: no license fees and no need to become a database company. Java DB is also great for use with Java applications and for Java application development, something that Sun generally likes to foster. Finally, Sun must provide support for

Java DB as it is included in other Sun products. Selling those support services "stand-alone" is a way for Sun to derive revenue from otherwise fixed costs and it allows our customers to use Java DB in production should they find, as we did, that it's the best database for their needs.

Q: What role does the community play in the development? How is it significant that Java DB is open source? What is the single greatest benefit and biggest challenge from being open source?

A: Public discussion by the community improves the thoroughness of feature proposals and the robustness of actual code. Many bugs are fixed as the community desk-checks submitted code. As awareness about Java DB grows, we expect that other companies will fund Derby development.

I would say the single greatest benefit of open source is the high quality of the product.

I would say the single biggest challenge is the time devoted to community involvement: e.g., responding to email and reviewing other people's code. For an individual contributor, this can easily consume a whole day per week. For a committer, this can consume two days per week.

Q: How significant is modularization for open source products?

A: I'm not sure I understand this question. Derby is componentized and designed in layers. I cannot say that this appreciably affects community involvement or release management. I am not competent to generalize these modest observations to other open source efforts.

Q: What is Java DB primary strategy to get new users: grow the overall SQL market or try to grab users from the competition? Are you mainly competing with other open source databases or closed source databases?

A: Sun's strategy is to enable customers to use the DB that best meets their needs, so really none of the above apply. For example, in addition to Java DB, Sun also offers support for PostgreSQL, ships both PostgreSQL and MySQL with Solaris, and was recently shipping Oracle with some of its servers. On the Derby user mailing list we have seen Derby draw developers (and their end-users) away from expensive and/or cumbersome products, both open source and proprietary.

It is too early to say whether Derby is growing the overall SQL market. Derby's platform independence may help here. Java DB certainly could help grow the Java user base.

Q: I heard that Java DB will be bundled with Java 1.6. Could you briefly explain how this came about and what effect you think this will have on the Java market and their usage of SQL will have? Will this take away from existing vendors or simply grow the overall SQL market?

A: The Java language and the original Cloudscape database grew up together. Cloudscape was the first database bundled on the Sun CDs which carried reference implementations of Java technologies. Bundling with the JDK is a natural next step. Bundling makes this free, small database handy to millions of developers. That, in turn, means that Java developers do not need to hand-roll their own persistence solutions. In a sense, this turns the following question on its head: "Why would I use an expensive, heavyweight SQL database to solve my small persistence problem?" The question becomes "Why wouldn't I use a database–it's free, lightweight, easy to master, and already on my desktop?" Bundled Java DB is useful to anyone who needs efficient collections and rollback after errors. Java DB has the potential to introduce databases into applications which previously dismissed SQL technology outright.

Q: What sort of market share does Java DB have in the overall and open source database market have according to your knowledge?

A: I don't know and this is very hard to measure with open source software. In Forrester's recently published "Open Source Database Wave", Forrester rated Derby a "strong performer". Given that Derby was virtually invisible even a year ago, this looks like great progress.

Q: What are the challenges with community cooperation from the perspective of Sun?

A: Sun faces the same challenges faced by any company committed to open source community development. I think the most awkward challenge is coordinating release trains: how do you align the community's release plans with those of your own products? You can't accelerate the community's judgment about when code is mature enough for release. All you can do is wait for the community to finish its work.

Q: What is your perception about the free for development and deployment stripped down versions provided all the major proprietary vendors that usually run under the

"Express" label? Are these version a reaction to open source or is this simply natural versioning of increasingly complex products?

A: Limited products address only one of the open source challenges: the $0 price tag. For instance, these packages don't enable software escrow or rapid put back of critical bug fixes. Many also have truly limited capabilities or are even hobbled in some way. From my vantage point in open source, these offerings look like brief transitional points on the longer market trajectory toward free and open source software: Customers don't want to pay for software, but they will pay for support and services.

Q: Is the database market being commoditized by open source databases?

A: I think that open source helps to drive price out of the equation. But I don't know how to evaluate claims about commoditization. To me that term implies that databases are interchangeable. I don't think they are. In addition, I think there's a lot of stickiness induced by the time it takes to learn the intricacies of a new database.

Q: Do you see potential for cooperation with other open source RDBMS? And if so in what areas? For example Dan Debrunner was present at the preliminary meeting to create an Open Source Database Consortium in November 2005.

A: Yes, I do see a lot of potential for cooperation. As I said above, I don't think databases are interchangeable. They tend to be optimized for different application domains and platforms and they tend to target different ends of the market.

Q: How significant is standards compliance for the database market in general and Java DB users in specific?

A: Standards compliance means a lot to developers. It significantly reduces the steep cost of mastering a new database. Derby follows the JDBC and ANSI SQL standards. We believe that non-standard APIs scare off new developers.

Q: What, if any, influence does the Java DB project have on the SQL standard process? Is there a need to change this process?

A: Derby does not sit on any standards bodies today. However, through the companies they work for, individual members of the Derby community can influence the standards.

So far, Derby has been able to live with the existing SQL standard and has not needed to influence its direction. Individuals, through their company representatives at ANSI,

have sought to clarify spec ambiguities. Derby has had some influence on the JDBC spec because the spec lead is a member of our community.

Q: Is there anything else you want to say about open source in the relation database market?

A: That's all.

## A.2 Firebird

### A.2.1 Holger Klemt

Q: Could you briefly introduce yourself and your role inside the Firebird project?

A: Member in the Firebird Foundation since 2002, organizer of the worldwide Firebird Conference since 2003, currently planning 2006 conference, Owner of HK-Software, major Products are different individual database applications based on firebird, interbase or ibm iseries/as400 for large customers, IBExpert Database Development Tools

Q: What are the primary features that set Firebird apart from other RDBMS products?

A: easy installation and usage, really no maintenance required when typical programming errors were not made. We have customers running the firebird server with their application since 4 years without any changes on maintenance. this is how a database server should do his job. The job called "Database Administrator" makes no sense, there is no need for this with modern database servers. The licensing allows full commercial usage without any licensing costs, so firebird is the number one choice for software development companies, who need a robust and fast database server with low or no maintenance.

Q: Could you give a brief history overview of Firebird?

A: a good history can be found here `http://www.cvalde.net/ibRoadmap.htm` and here for the future `http://www.firebirdsql.org/devel/engine/roadmap2006.html`

Q: Could you explain how the Firebird project works? How is the code developed? How many developers are part of the project and what are the main companies that employ them?

A: here is a nice overview: `http://www.firebirdsql.org/index.php?op=devel&sub=engine`

Q: Why did you join the Firebird project?

A: in my eyes the best choice not only among the open source database systems. it never disappointed me or a customer, when they were able to understand which special technology is used and how you have to handle it. a lot of large companies use the server together with our software worldwide

Q: The Firebird project was the result of the first large open sourcing of a previously closed source database. Why do you think that Borland decided to continue their closed source development? What sort of cooperation is there still with Borland?

A: since borland is no longer responsible for interbase, we will see what the new Devco will change. typical borland guys (team members) try to ignore firebird, but this will not work, especially not typical delphi programmers.

Q: How has your involvement in Firebird project affected your professional career?

A: since more and more users do use firebird, we also sell more and more licenses of our software ibexpert.

Q: What are the types of business models employed by commercial proponents around Firebird? What are the primary sources of income for these companies? Are there other significant (emerging) revenue streams (certification and books perhaps)? Are there any other numbers you can share about these companies? Like what percentage is invested into R&D, marketing/sales, community relations.

A: hard to say, we are very involved in adding new technologies in our product, that is missing in the firebird server. since we sell standard software (ibexpert), do trainings based on firebird and ibexpert and also sell support to customers worldwide, we are very satisfied with the current market development.

Q: What role does the community play in the development? How is it significant that Firebird is open source? What is the single greatest benefit and biggest challenge from being open source?

A: the biggest advantage for customers is that they often start their work with firebird because it is free (not only free for gpl applications like a swedish database system). After spending some time using firebird, most users will not change again to another database.

Q: How significant is modularization for open source products?

A: very important, since there is not everybody interested in developing the core engine, the odbc driver, the .net driver, the java driver etc.

Q: Is there anything that needed to be changed at the source level due the open sourcing? For example were there any proprietary pieces that needed to be replaced or simplifications in the build process that had to be done?

A: most of this is already done, especially the changes to c++

Q: Why did the Firebird project decide to rewrite the source code to C++?

A: must be answered by core developers

Q: What is the significance that Firebird uncovered an old backdoor in the interbase source code? What sort of reactions resulted from this discovery?

A: this shows that open source works. such a backdoor is really not a good thing but as far as i know a lot of closed source software products do have such things

Q: What was Borland's initial, and that of todays in Firebird involved companies, strategy for building a community around Firebird.

A: borland ignores the existence of their open source product since the time right after opening the source. it seems that they still think that a database server or client license can be sold in the future. i remember that a lot of years ago it was possible to buy tcp/ip network drivers, these products do no longer exists. i think it will be the same with typical database servers. they are just available for your operating system or not. the firebird community is very helpful to new users, so we think that a lot of people with low level and high of experience will see that the firebird server will be the best choice for their needs

Q: What is Firebird primary strategy to get new users: grow the overall SQL market or try to grab users from the competition? Are you mainly competing with other open source databases or closed source databases?

A: firebird has no big marketing power because there is no venture capital that helps to create more visibility. on the other hand, there is also no company that will be

under pressure when their business model using venture capital will have not the expected results.

Q: What sort of market share does Firebird have in the overall and open source database market have according to your knowledge?

A: from our perspective, we know that all in all about 50000 have downloaded ibexpert until now. a new version will be downloaded at least 30000 times in typically 3 month. Most of them use the free personal edition, but a lot of them are also paying customers. when i think that not all people who work with firebird do have ibexpert and a lot of people of these are responsible that other people in their company or at their customers sites do use the firebird server, i expect that there are more than 1 million user worldwide who work with the firebird server in their daily work. we have a lot of interesting names in our database like, lufthansa, airbus, motorola, nokia, german telekom, siemens, bayer, us army, french army, ....

Q: How do you feel about the commercial involvement in Firebird? What is the basis for this cooperation? What are the challenges and requirements from the perspective of the community?

A: since the commercial companies like HK-Software or IBPhoenix are also responsible for giving the firebird project a better visibility, so it is a give and take relationship. most of these companies also directly support the firebird foundation as a member.

Q: What is your perception about the free for development and deployment stripped down versions provided all the major proprietary vendors that usually run under the "Express" label? Are these version a reaction to open source or is this simply natural versioning of increasingly complex products?

A: this is only a temporary way, most of these products will only give a chance for experienced users to get small versions for free. Since even an open source software like the compiere erp system, which is created based on oracle, does not work with oracle express version because a missing java support module, this is more crippelware. in almost all license agreements you also find a hint that you have to stop work with express edition, when the manufacturer wants to stop this product line. we will see what happens ....

Q: Is the database market being commoditized by open source databases?

A: yes, it is interesting to note that in the past people only considered what database product to by from what vendor and that today many more applications are based

on professional databases, for example the avg antivirus software uses the firebird server for its server installation.

Q: Do you see potential for cooperation with other open source RDBMS? And if so in what areas? For example several Firebird project members where present at the preliminary meeting to create an Open Source Database Consortium in November 2005.

A: after the first meeting where i also have been in frankfurt, there was no communication reaching me. i think as in most other projects, each project need their own supporters, a typical postgresql guy will support his team, a firebird guy will support his team. like in soccer, there is a market for every soccer team at least in their own town. In the database market, there are worldwide towns with fans for different ways how to do it.

Q: How significant is standards compliance for the database market in general and Firebird users in specific?

A: very significant, i found that regarding the mandatory core sql standards, firebird and postgresql have almost the same level of supported functionality, and i know that for example working with firebird and ibm db2 at the same time is not a big deal. almost all statements work on both systems in the same way.

Q: What, if any, influence does the Firebird project have on the SQL standard process? Is there a need to change this process?

A: as far as i see there is no influence until now

Q: Is there anything else you want to say about open source in the relation database market?

A: open source is the future of database systems. have a look at database newsgroups, where still now a lot of people find very strange named files that were created by a closed source database system which does no longer exists. without having the source code it is almost impossible to write an import job for these files, when you have access to the source code, it is a much easier job.

## A.3 MaxDB

### A.3.1 Ulf Wendel

Q: Could you briefly introduce yourself and your role inside the MaxDB project?

A: I am working for MySQL as a Support Manager MaxDB. The job covers support, training, consulting and pre-sales tasks.

Q: What are the primary features that set MaxDB apart from other RDBMS products?

A: MaxDB is a heavy-duty enterprise database. The database management system is SAP-certified. MaxDB has been designed with the goal of archiving low Total Costs of Ownership in mind.

MaxDB has one of lowest TCO among all SAP-certified databases. MaxDB is available on all major platforms. You can choose whatever platform is best for your.

Graphical user interfaces and wizards help you to perform all kinds of configuration and administration tasks. If you prefer a command line interface, it's there as well.

The storage management of MaxDB is completely automated. No reorganization is needed, no fragmentation can happen. This guaranties continuous operation in demand 24x7 environments with large number of users and high workloads as no scheduled downtimes are required.

Backup and Recovery functionality is built in and can be smoothly integrated into third-party solutions. High availability solutions are possible using cluster technology.

Q: Could you give a brief history overview of MaxDB?

A: The database development started in 1977 as a research project at the Technical University of Berlin . In the early 80s it became a database product that subsequently was owned by Nixdorf, Siemens Nixdorf, Software AG and today by SAP AG. Along this line it has been named VDN, Reflex, Supra 2, DDB/4, Entire SQL-DB-Server and ADABAS.

In 1997 SAP took over the software from Software AG and renamed it to SAP DB. Since October 2000 SAP DB sources additionally were released as open source under the GNU General Public License.

In 2003, SAP and MySQL concluded a partnership and development cooperation agreement. As a result, SAP's database system SAP DB has been delivered under the name of MaxDB by MySQL since the release of version 7.5 (November 2003).

Q: Could you explain how the MaxDB project works? How is the code developed? How many developers are part of the project and what are the main companies that employ them?

A: The former SAP DB development team at SAP AG is responsible, now as before, for developing and supporting MaxDB. MySQL AB cooperates closely with the MaxDB team at SAP around delivering improvements to the MaxDB product. Both SAP AG and MySQL AB handle the sale and distribution of MaxDB. The advancement of MaxDB and the MySQL Server leverages synergies that benefit both product lines.

MaxDB is subjected to SAP AG's complete quality assurance process before it is shipped with SAP solutions or provided as a download from the MySQL site.

SAP AG employs some 70 developers to continue the development of the technology flagship MaxDB.

Q: What is the relationship between MySQL AB and SAP on the MaxDB project?

A: MaxDB is the new name of a database management system formerly called SAP DB. In 2003 SAP AG and MySQL AB joined a partnership and re-branded the database system to MaxDB. The development of MaxDB has continued since then as it was done before-through the SAP developer team.

MySQL AB cooperates closely with the MaxDB team at SAP around delivering improvements to the MaxDB product. Joint efforts include development of new native drivers to enable more efficient usage of MaxDB in the Open Source community, and improvement of documentation to expand the MaxDB user base. Interoperability features between MySQL and MaxDB database also are seen as important. For example, the new MaxDB Synchronization Manager supports data synchronization from MaxDB to MySQL.

MySQL AB offers a complete portfolio of Professional Services for MaxDB.

Q: Why did you join the MaxDB project?

A: I was offered a challenging job :-) . The questions should be a different: why to choose MaxDB over any other database?

Q: The MaxDB project was the result of the open sourcing of a previously closed source database. What has changed for existing customers as a result of this? Has there been a reaction by these customers?

A: Existing and new customers have profited from the opening.

For existing customers not much has changed. SAP AG has continued the MaxDB development just as before the opening. End of life warranties and service offerings have been remained the same when the product went Open Source in October 2000.

New customer have been given the opportunity to improve MaxDB according to their need. Customers, who respect the regulations of the GPL, can use the database system without paying any license fees.

When SAP AG and MySQL AB re-branded MaxDB in 2003, parts of the community feared this could be the end of MaxDB. But this is not the case. Just the opposite is true: MySQL AB started to offer Professional Services to existing and new customer.

Q: Could you please explain the MaxDB business model? What sort of benefit does MaxDB gain from being open source?

A: Through the open source distribution MaxDB gets:

- more users that are familiar with the product.
- customers can try out the product easily as free SW, docs, forums etc. lowers the barrier to try the product out
- community gives feedback on the product through bug-reports, through discussions in lists and forums, which all help continuously to improve the product
- open source branding gives SAP customers the feeling that they can rely on the product being cost efficient (compared to for instance Oracle) also for SAP Applications now and in the future

Q: How does MaxDB relate to MySQL?

A: MySQL offers a variety of products, mainly the MySQL database server.

Through the co-operation with SAP, MySQL AB provides MaxDB both to customers and to open source community, as complementary offering: a SAP certified enterprise database capable of running heavy duty applications like the SAP Business Suite.

Q: When I was talking to people at Oracle about open source databases they noted that in their opinion dual licensing in the MySQL AB model is not community driven, since there are really close to no code contributions. What do you answer to that?

A: The main value community provides us is through excellent bug reports and improvement suggestions. Due to the code being open, these reports and suggestions can go into a very complex level of detail - i.e. provide an even huger value.

Q: What role does the community play in the development? How is it significant that MaxDB is open source? What is the single greatest benefit and biggest challenge from being open source?

A: See previous.

Biggest challenge is perhaps to go through and process the huge amount of community input we get, but this is a very pleasant and positive "challenge" to have.

There are two examples of daily benefits:

- user reports - you need feedback and people reflecting your ideas, e.g. on Eventing
- popularity - people tell others about their experiences

Q: How significant is modularization for open source products even if they are entirely development in house like MaxDB?

A: MaxDB has been using a modular design from the very beginning. This is just a matter of a good and clean coding style. This makes a software more maintainable. There's no big difference between doing this for your in house developers or for external developers. It is simply a must for a quality product like MaxDB.

Q: Is there anything that needed to be changed at the source level due the open sourcing? For example were there any proprietary pieces that needed to be replaced or simplifications in the build process that had to be done?

A: No major changes had to be done.

Q: What is MySQL AB/SAP strategy for building a community around MaxDB.

A: Use MYSQL's popularity (mainly the website) and familiar community tools (mainly being open about everything we do and driving user-involvement in lists, forums and at events), to spread from the MySQL product also to this complementary database offering.

Q: What is MaxDB primary strategy to get new users: grow the overall SQL market or try to grab users from the competition? Are you mainly competing with other open source databases or closed source databases?

A: We do all of the above. No clear division noticed between getting new users / turning users from other products, or between competing with open source DBs (mainly MySQL and Postgres) and Closed source DBs (Oracle, DB2, Informix, Sybase, MSSQL). Pretty even.

Q: What sort of market share does MaxDB have in the overall and open source database market have according to your knowledge?

A: In the SAP market, MaxDB has a market share of about 4%. We do not have any figures for the Open Source market, but we can tell you that MaxDB has some 50.000 downloads per month.

Q: What are the challenges with community cooperation from the perspective of MySQL AB?

A: To get users to contribute in a way that rewards both them and us at the same time. Logically, the more they feel rewarded themselves, the more actively they are ready to put in effort to help us.

Q: What is your perception about the free for development and deployment stripped down versions provided all the major proprietary vendors that usually run under the "Express" label? Are these version a reaction to open source or is this simply natural versioning of increasingly complex products?

A: Yes, I think it is a reaction to them noticing that they are losing developers to Open Source products, who develop on open source, and then do NOT upgrade (as expected) to proprietary DBs when they deploy but stay on the Open Source platform. By offering restricted versions for free, they hope to attract the developers early on to the proprietary platforms.

Some comments about these Express versions.

The "free" databases typically impose the following restrictions:

- Restricted to Single Machine
- Restricted by # of CPUs
- Restricted by Amount of RAM
- Restricted by Database Size

- Restricted by Replication (Scale-out)
- Restricted by Web (Hosted) deployment
- Restricted by Lack of Technical Support offerings

The impact of these restrictions to users are often:

- NOT allowed to scale-up or scale-out their applications due to the Deployment and Technical Restrictions
- NOT allowed to deploy high-performance hardware required to provide a positive user experience.
- NOT allowed to deploy databases applications larger than 4-5 GB limiting the potential of the application
- NOT allowed to deploy Data Warehouse of Data Mart applications due to the DB Size limitation.
- NOT allowed to scale beyond a certain number of users due to the RAM/CPU/Disk limitations
- NOT allowed to take advantage of parallel operations such parallel queries, loading, and partition scans due to the single CPU limitation.
- NOT allowed to receive Production-level Technical or Consultative Support required to deploy business or mission-critical applications.
- NOT allowed to choose an operating system that reduces costs and increases reliability and performance

Q: Is the database market being commoditized by open source databases?

A: Yes.

Q: Do you see potential for cooperation with other open source RDBMS?

A: Sure. I think along different kind of standardization and around Open source licensing issues co-operation comes naturally.

Q: And if so in what areas? For example Patrik Bachman represented the MaxDB project at the preliminary meeting to create an Open Source Database Consortium in November 2005.

A: See above. Another co-operation area for the future could be around DB tools.

Also more deeper co-operation can be interesting, if we just find a good win-win match for both parties. Note, that we are co-operating with other Open Source database vendors already, with Oracle and Solid around them providing Storage Engines for MySQL. Here, there is a good win-win match, and I see no reason why similar co-operation could not start with other Open Source DB players.

Q: How significant is standards compliance for the database market in general and MaxDB users in specific?

A: MySQL has always aimed at being very ANSI compliant, and we believe standards are a good way of ensuring the product behaves for new users as they would expect.

For the *biggest* proprietary players I don't think standards matter too much, as with a given size and user "lock-in", interoperability between your syntax to others might not be so relevant.

MaxDB customers are mostly in the SAP worlds, and they mostly care about their DB behaving as expected with the SAP Applications. In the open source worlds, we have done some efforts to get MySQL and MaxDB closer in terms of syntax, but still big differences do exist.

The usage of MaxDB can be split into two markets: MaxDB for SAP and MaxDB for Non-SAP environments. SAP customers do not ask for SQL standard compliance in general. Their question is: "Does it work with SAP solution, I'm planning to use?" For Non-SAP environments we rarely get requests on standard compliance.

MaxDB is compliant to ANSI SQL-92 (entry level).

Q: What, if any, influence does the MaxDB project have on the SQL standard process? Is there a need to change this process?

A: I have no insight into the SQL Standard Process. However, we have persons inside MySQL who do track (and might even participate in) the standards progress closely, and they keep our development aware of what is going on.

Q: Is there anything else you want to say about open source in the relation database market?

## A.4 MySQL

### A.4.1 Kaj Arnö

Q: Could you briefly explain your role inside MySQL AB?

A: I joined MySQL in 2001. As VP Community Relations, a position I have had since September 2005, my responsibility is the growth, happiness and activity of the MySQL community.

Q: The bulk of MySQL AB's developers work from home. What sort of benefit does this give the company? What sort of challenges does this create and how are those met inside MySQL AB? What sort of benefits and limitations do you see in such an organization of labor for other companies (in other markets and industries)?

A: Benefits: We can recruit people based on their skills, regardless of where they work. And they can continue to work without relocating. They can work without daily communes. They can work without being interrupted. We don't pay big office rents.

Challenges: Working without seeing your colleagues more than a few times a year. Need for travel increases, so does its cost. Time zones makes Europeans work late nights and Americans stand up early.

Q: What role does the community play in the development? How is it significant that MySQL is open source? What is the single greatest benefit and biggest challenge from being open source?

A: The community is the fundament, without which MySQL wouldn't exist. It feedbacks us with requirements and with bug reports. Benefit: The community wouldn't exist if MySQL weren't Open Source. Challenge: Finding the best differentiator between free-of-charge and commercial, i.e. the borderline between what you get for free and what you have to pay for.

Q: When I was talking to people at Oracle about open source databases they noted that in their opinion MySQL is not community driven, since there are really close to no code contributions. What do you answer to that? How is this related to the various pluggable interfaces that are emerging in MySQL?

A: Our code is mostly written by MySQLers. And through our ownership of the copyright to our code, we have the Dual Licensing business model. That said, we

are simplifying our contribution policy and have for instance seen new storage engines written by the community, such as the PBXT storage engine written by Paul McCullough in Hamburg, Germany.

Q: What are the primary sources of income for MySQL AB? Also what is the ratio on the income side between support contracts and licensing. Are there other significant (emerging) revenue streams (certification and books perhaps)?

A: Our largest revenue stream is still commercial licensing, but our most growing revenue stream is MySQL Network, our subscription based offering for the enterprise sector. Other significant sources of income are training and consulting.

Q: What is MySQL AB's primary strategy to get new customers: grow the overall SQL market or try to grab customers from the competition? Are you mainly competing with other open source databases or closed source databases?

A: Of course, we see other databases in competitive situations, but the growth is mainly based on overall database market growth.

Q: What is MySQL AB's perception about the free "for development and deployment" stripped down versions provided all the major proprietary vendors that usually run under the "Express" label? Are these version a reaction to open source or is this simply natural versioning of increasingly complex products?

A: We believe the market sees a distinction between "free of cost" databases (which have strings attached and where you can expect to one day pay for your lunch) and databases released as free software. Companies releasing the software under GPL have already freed the software of the strings attached. Proprietary database vendors might believe "Express" versions can address the new constellation which the Open Source databases have created, but I do think there is a fundamental distinction between freedom and no-cost.

Q: Is the database market being commoditized by open source databases?

A: Yes, this is a clear trend which has already been in place for a while.

Q: MySQL has initially only implemented a very small subset of the standard. A fair number of non standard extensions were also made like LIMIT, REPLACE and the ENUM data type. How significant is standards compliance for the database market in general and MySQL users in specific?

A: MySQL started by solving a subset of the generalist database user needs, and has for each release complied to the standard to a larger and larger degree. We strive for full standard compliance.

Standardisation is important for the end user, because they don't want to be locked in to individual products. Swapping databases should be as easy as possible.

Q: What, if any, influence does MySQL AB have on the SQL standard process? Is there a need to change this process?

A: We don't yet participate in the standards process except indirectly, by which I mean that we see the LIMIT statement appear in other databases without any work in a standards body.

Over time, we can see MySQL participate more in standardisation processes. It's too early for us to tell whether that process should be optimised in some way.

### A.4.2 Lenz Grimmer

Q: Could you briefly explain your role inside MySQL AB?

A: I am a member of the Community Relations Team, where I work on improving our interaction and relationship with the community of MySQL users in various ways. I report to Kaj Arnö, our VP Community Relations (whom you interviewed as well).

Q: What are the primary features that set MySQL apart from other RDBMS products?

A:

- Performance, Reliability, Ease of Use (not necessarily in that order)
- MySQL is Open Source, not just free as in beer!
- A very active and vocal community of users and a large software ecosystem around the MySQL Server (both OSS and commercial)

Q: Could you give a brief history overview of MySQL?

A: Of the product or the company? For the product, see the Change History in our manual:

`http://dev.mysql.com/doc/refman/4.1/en/news.html` `http://dev.mysql.com/doc/refman/5.0/en/news.html`

For Company history, please see `http://en.wikipedia.org/wiki/MySQL_AB` and the German page at `http://de.wikipedia.org/wiki/MySQL_AB` - our web site at `http://www.mysql.com/company/` also has some more info. Let me know if you have any specific questions about the company history not covered there.

Q: Could you please explain the MySQL AB business model?

A: Quoting the link mentioned above:

Revenue Model

MySQL AB has three main sources of revenue:

1. Online support and subscription services sold globally over the MySQL.com website to all users of the MySQL server.
2. Sales of commercial MySQL licenses to users and developers of software products and of products that contain software.
3. Franchise of MySQL products and services under the MySQL brand to value-added partners.

We provide our customers and partners with support services they can depend on, consulting services, training programs, and more.

Q: What sort of benefit does MySQL gain from being open source?

A: Primarily ubiquity - it's easy to deploy and use MySQL. The OSS aspect puts confidence in our product and it allowed us to build up the community around it that we have nowadays. We would not have created the same ecosystem around MySQL had it been a closed-source product. Being Open Source also helps to improve the quality and security - anybody is welcome and encouraged to look at the code.

Q: The bulk of MySQL AB's developers work from home. What sort of benefit does this give the company?

A: It allows us to employ people across the globe without having to require them to relocate to another location, which might not be possible for some. So it's easier for us to hire talent that we otherwise would not have gotten. Note that it's not only developers that work from home - the concept goes through the entire company. For

example, this a very important asset for our support team, as we need to have a well distributed coverage in every time zone.

This article in Fortune Magazine covers a few other aspects: `http://money.cnn.com/2006/05/31/magazines/fortune/mysql_greatteams_fortune/`

We may actually save some money by not having to rent large facilities, but I can't tell if that is a significant aspect. Money is spent elsewhere, for people to maintain their work environment, or to fly employees to a central location for meeting and discussing issues.

You might also want to read Kristian Koehntopp's recent blog entry about this topic (in German):

`http://blog.koehntopp.de/archives/1318-Erfahrungen-mit-Nonoffice.html`

Q: What sort of challenges does this create and how are those met inside MySQL AB?

A: The main challenge is of course making sure that work actually gets done. This is monitored by defining fixed goals one has to achieve and by regular reporting and meeting over phone or IRC. But every team has a slightly different approach to this - in the end it's about getting the work done that has been agreed upon beforehand.

Another challenge is the information flow - how can we make sure that everybody is in the loop about things that affect his work? How can we avoid information overflow? To keep other departments posted on what's going on elsewhere, we send out monthly "State of <department>" emails, that summarize the important events and activities inside that team. We also have a regular conference call with our CEO, where everybody dials in to listen to Marten speak, while one can post questions and comments on IRC that Marten then elaborates on.

So the overall communication overhead may be higher than in traditional firms, but it leads to very structured and well-defined processes, especially if the company grows along with this distributed nature.

Also, integrating new colleagues is a bit more challenging. It's very important to actively involve them from the very beginning and aid them properly to find their way around, otherwise there is a high chance they get lost and frustrated at the very beginning, if they are not that experienced with working from home all by themselves. You have to be made for working in this kind of environment, but it's a great help if

you have the opportunity to physically meet with the people you will be working on at the beginning.

Q: What sort of benefits and limitations do you see in such an organization of labor for other companies (in other markets and industries)?

A: Probably not all companies are suitable for being virtualized like that. It's also important that it's much more difficult to turn an existing company culture around (by introducing the concept of home offices), than growing with that concept in mind from the very beginning. It depends on what work needs to get done, not every work can be distributed like that. But for most work that requires a computer and an Internet connection (e.g. Software Development, CAD, writing documents) or a phone (e.g. support, sales) this model works quite well.

Q: How significant is modularization for open source products even if they are entirely development in house like MySQL?

A: Very significant, if you ever want to extend or improve your product. A big monolithic blob becomes unmaintainable at some point, be it a closed in-house product or an OSS application. The better modularized the product is, the easier it is for others to get going with it - they don't have to grasp the full picture and can focus on working with a small component only. Our pluggable storage engine architecture is one popular example of that.

Q: What role does the community play in the development?

A: The user community provides us with the feedback we need to stay on track with our development and is a very valuable asset to spot bugs and regressions on our side. They request the features we will eventually implement. Users can become very creative in how they utilize MySQL, this gives us very important insight into how we can further improve our product.

We also receive patches and code contributions from the community, but probably not at the same rate as other OSS projects do. It takes a while to get familiar with the MySQL code base and the incorporation of external code contributions requires a bit of legalese (as we need to maintain the full copyright on the entire code base).

Q: How is it significant that MySQL is open source?

A: I think it's one of the most important assets and is an essential component that actually brought us to where we are today. MySQL would not have had the same success if it had been a closed-source product.

Q: What is the single greatest benefit and biggest challenge from being open source?

A: The greatest benefit is probably the large ecosystem that has evolved around MySQL over the years. The OSS aspect of MySQL makes sure that it easy to combine it with other OSS software. Like Linux, MySQL is a very low level component of an OSS application stack - quite many applications build on top of it. So like Linux, for many MySQL has become a commodity OSS application.

The biggest challenge is probably maintaining and extending this ecosystem and not to lose track of what you are doing at some point. To a large extent, OSS is a about community. They will vote with their feet, if you don't listen to their feedback or concerns.

Q: Do you think that in 2-3 years the storage engine usage pattern will have diversified, from currently two storage engines (MyISAM and InnoDB) to a situation where 3 or more storage engines are in common use?

A: Yes, this is already happening!

We already observe a significant adoption of MySQL Cluster as a High Availability storage engine. Alcatel and Nokia are two high profile customers in that area, and we have several other customers, mostly in the TelCo business. See `http://www.mysql.com/products/database/cluster/` for more info and `http://www.mysql.com/why-mysql/case-studies/` for some case studies.

And we also notice that there is a growing number of 3rd-party storage engines that have not been developed by MySQL. SolidDB and PBXT are probably the two most prominent examples here - once they have matured enough, they will most likely find their use cases in which they would be preferred over InnoDB and MyISAM. By that time, our own Falcon Storage Engine will also be available in production, so this is going to be an exciting coopetition.

Q: When I was talking to people at Oracle about open source databases they noted that in their opinion MySQL is not community driven, since there are really close to no code contributions. What do you answer to that?

A: Community does not necessarily drive an OSS project via code contributions only. In fact, most OSS projects have a very closed circle of developers, it's all the other stuff that community does for a project that makes it succeed or fail.

But we are planning on making it easier for others to contribute code, currently the legal hassle about the copyright assignment is a bit of a stumbling block here. Once

this has been worked out, we will actively encourage the community to contribute more.

Q: How is this related to the various pluggable interfaces that are emerging in MySQL?

A: Closely related, of course. The entire plug in architecture that we are working on (which is not limited to storage engines only) is an important step into further opening up our development processes and fostering the ubiquity of MySQL.

Q: There was a huge uproar in the community due to changing the license on the client libraries from the more liberal LGPL to the more restrictive GPL. As a result MySQL AB put in place an exception clause for FLOSS projects. Given the fact that MySQL AB stresses the importance of the community how was it possible that this problem was not foreseen beforehand? What has MySQL AB learned from this for the future?

A: I would assume we simply underestimated the negative reactions of the community about that - we should have made our intentions more clear beforehand instead of silently changing the license. I personally think that the current license (GPL plus FLOSS Exception) is actually more OSS-friendly than how it used to be before. It helped to have a clearer distinction between the commercial and OSS side of including and distributing MySQL.

Q: Was the acquisition of InnoDB directed at getting a share of MySQL AB's increasing success? Or rather to stall this progress?

A: I can't comment on Oracle's motivations, you would have to ask them about their intentions here :)

Q: What sort of communication has emerged now that Oracle and MySQL AB are actually partners due to the InnoDB licensing agreement?

A: It pretty much "business as usual" for us. We continue to work with the InnoDB developers as before and they are responsive and helpful and continue to improve on InnoDB.

Q: What are the primary sources of income for MySQL AB? Also what is the ratio on the income side between support contracts and licensing. Are there other significant (emerging) revenue streams (certification and books perhaps)?

A: Up to the end of 2004 the two major source of revenue were license sales and Professional Services (Support and Training). Other activities such as Consulting, Certification, Partnerships did not really contribute much. This has now changed with

the MySQL Network Subscription model (`http://mysql.com/network`), which has by now become our main source of revenue. However, license sales and other individual services are still contributing their share, too.

Q: Are there any other numbers you can share about MySQL AB? Like what percentage is invested into R&D, marketing/sales, community relations.

A: Sorry, I personally can't as I have no insight into these. You might want to ask Kaj if he could give you a more detailed insight.

Q: What is MySQL AB's primary strategy to get new customers: grow the overall SQL market or try to grab customers from the competition? Are you mainly competing with other open source databases or closed source databases?

A: We see ourself as a commodity database that actually serves a niche that has not been properly occupied by any other players so far. The overall database market is expanding rapidly, which leaves enough room for us to grow and participate.

Especially a majority of the emerging generation of Web 2.0 applications are based on MySQL. (See Tim O'Reilly's definition of "What is Web 2.0?": `http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html`)

We have listed a few case studies of MySQL Web 2.0 customers here (scroll down to the bottom): `http://www.mysql.com/customers/`

"Grabbing" customers from other vendors is much harder, as it's always painful to switch to another infrastructure without good reason. But we of course support customers that are willing to do the switch as good as we can. So we do not want to directly compete with the commercial vendors, we rather compliment their offerings. We also do not see other OSS databases as competition. In fact, we share the same goal, which is to replace closed source DBMSes with OSS alternatives.

Q: What sort of market share does MySQL have in the overall and open source database market have according to your knowledge?

A: This is very hard for me to estimate, especially since we don't have easy ways of determining the overall installed base of MySQL servers (as we can not simply count licenses sold).

Q: What are the challenges with community cooperation from the perspective of MySQL AB?

A: The challenge is to stay on the thin red line between being a commercial entity that has to sustain itself and generate revenue, without neglecting or treading on the non-paying community user base.

Q: What is MySQL AB's perception about the free "for development and deployment" stripped down versions provided all the major proprietary vendors that usually run under the "Express" label? Are these version a reaction to open source or is this simply natural versioning of increasingly complex products?

A: These crippled versions are very likely a direct reaction of the increasing threat by Open Source products. However, these databases are usually quite limited in their functionality and only function as a hook to lure the users into buying more expensive licenses. We believe that users prefer truly free products without artificial limitations and strings attached and to our observations the crippled DBs don't really play an important role for new deployments.

Q: Is the database market being commoditized by open source databases?

A: We certainly hope so :)

Q: Does MySQL see potential for cooperation with other open source RDBMS? And if so in what areas? For example MySQL AB organized and sponsored a first preliminary meeting to create an Open Source Database Consortium back in November 2005.

A: There is potential, especially when it comes to advocating Open Source Software in general. On the technical level, there might not be too many similarities that really provide room for collaboration. But we do maintain relationships to the other OSS projects. Diversity is good and each of these OSS RDBMSes has their own merits and downsides.

Q: MySQL has initially only implemented a very small subset of the standard. A fair number of non standard extensions were also made like LIMIT, REPLACE and the ENUM data type. How significant is standards compliance for the database market in general and MySQL users in specific?

A: When it comes to implementing new functionality, MySQL nowadays works hard on complying to the relevant standards. It is important for us to not deviate, but we're not holistic about it. It's the users who sometimes dictate what features they want (e.g. the LIMIT clause). But the overall goal is to get rid of the "gotchas" MySQL still has and to adhere to established standards, were possible.

Q: What, if any, influence does MySQL AB have on the SQL standard process? Is there a need to change this process?

A: To my knowledge, we're not currently actively involved in the SQL standardization process and I don't think we see a need to change it. However, we observe the progress closely and keep an eye on what's coming up.

Q: Is there anything else you want to say about open source in the relation database market?

A: OSS in the RDBMS market has already started to become a commodity. OSS databases are already "good enough" for the majority of purposes. So it's very likely that this segment will experience the same than what happened on the OS sector, where OSS operating systems like Linux and *BSD have already become a commodity.

## A.5 PostgreSQL

### A.5.1 Gavin Sherry

Q: Could you briefly introduce yourself and your role inside the PostgreSQL project?

A: My name is Gavin Sherry. I am a programmer from Australia. I have no formal computer science or engineering training. I run my own PostgreSQL consulting company and provider training, advice, engineering and administration services to Fujitsu and some of Australia's biggest companies. I've been involved with PostgreSQL for 7 years.

Q: What are the primary features that set PostgreSQL apart from other RDBMS products?

A: It is open source. It has an open development process – there is extremely little private discussion.

It has several technical features which set it apart: query rewriting, non-overwriting MVCC, point in time recovery, a wide variety of procedural languages and much more.

Q: Could you give a brief history overview of PostgreSQL?

A: `http://www.postgresql.org/docs/8.1/interactive/history.html`

Q: Could you explain how the PostgreSQL project works? How is the code developed? How many developers are part of the project and what are the main companies that employ them?

A: There are roughly two hundred contributors over a given development cycle. Less than 10 of those are actively coding on a daily basis.

Q: Why did you join the PostgreSQL project?

A: I was attracted by the quality of the code and the software itself – as well as the people involved.

Q: How has your involvement in PostgreSQL project affected your professional career?

A: I work only on PostgreSQL related projects.

Q: What are the types of business models employed by commercial proponents around PostgreSQL?

Services based, dual licensed, 'cripple-ware' (open source/free version, proprietary/-costly full functionality version), and subscription based models (a-la red hat).

Q: How significant is modularization for open source products?

A: Not sure I understand this question.

Q: What role does the community play in the development? How is it significant that PostgreSQL is open source? What is the single greatest benefit and biggest challenge from being open source?

A: PostgreSQL is its community. It would not exist if it was not open source.

Being open source makes the development/testing/release processes more efficient and fun. The draw back is that work has to happen at the pace of the developers, not that dictated by management.

Q: What is PostgreSQL primary strategy to get new users: grow the overall SQL market or try to grab users from the competition? Are you mainly competing with other open source databases or closed source databases?

A: There isn't a primary strategy. We try and get the message out through case studies, articles, tutorials, training, etc. In reality, I believe we attract people through merit and word of mouth.

Q: What sort of market share does PostgreSQL have in the overall and open source database market have according to your knowledge?

A: I believe PostgreSQL to be the 5th most popular relational database in the world. I have no evidence for it. It is the second most popular open source relational database.

Q: How do you feel about the commercial involvement in the PostgreSQL, specifically about the closed source forks? What is the basis for this cooperation? What are the challenges and requirements from the perspective of the community?

A: I have nothing wrong with closing a fork or using some commercialisation mechanism to make money out of PostgreSQL.

Ultimately, however, I do not believe closing PostgreSQL and licensing it is a viable strategy. The pace of development in the community is such that companies spent too much time managing their fork and then they fall behind. Users do not want to downgrade to a commercial version.

Q: What is your perception about the free for development and deployment stripped down versions provided all the major proprietary vendors that usually run under the "Express" label? Are these version a reaction to open source or is this simply natural versioning of increasingly complex products?

A: This signifies just how concerned the closed source database vendors are about open source. The thing is, the express editions are not equivalent to us: we are not crippled (CPU, memory, data, support limits). Why would you even use one of these editions in production?

Q: Is the database market being commoditized by open source databases?

A: I'm not sure. Probably. In reality, .NET and JDBC are making the differences between databases irrelevant.

Q: Do you see potential for cooperation with other open source RDBMS? And if so in what areas? For example several PostgreSQL project members where present at the preliminary meeting to create an Open Source Database Consortium in November 2005.

A: It's still unclear what can be gained by that. I would like to see a basic web site which lists objectively true information about open source databases – the name, who develops it, its history, the target/market. Something like open source.org does with licenses.

Q: How significant is standards compliance for the database market in general and PostgreSQL users in specific?

A: The standard(s) give guidance but only DB2 has very good spec compliance.

Q: What, if any, influence does the PostgreSQL project have on the SQL standard process? Is there a need to change this process?

A: None. I don't know that we want to have any influence.

Q: Is there anything else you want to say about open source in the relation database market?

A: No that I have time for.. sorry :P

### A.5.2 Josh Berkus

Q: Could you briefly introduce yourself and your role inside the PostgreSQL project?

I'm Josh Berkus. I'm on the "Core Team" which is a steering committee for PostgreSQL, concerned primarily with releases. The main things I do for PostgreSQL are:

1. Public/Press Relations
2. handling our corporate sponsors
3. performance testing

Q: What are the primary features that set PostgreSQL apart from other RDBMS products?

A: Extensibility: you can very easily add your own functionality to the database.

Reliability: according to several sources, we are the most reliable, secure, and ACID-compliant open source DBMS.

License/Community: unlike other OS-DBMSes, we are under the liberal BSD license which permits commercialization. This is because we are an open community and are not controlled by a single company.

High-End Features: of the OS-DBMSes, we have the feature set and scalability closest to advanced proprietary DBMSes like Oracle and DB2.

Q: Could you give a brief history overview of PostgreSQL?

A: `http://www.postgresql.org/about/history`

Q: Could you explain how the PostgreSQL project works? How is the code developed?

A: An open association of developers, some of them employed to work on PostgreSQL and many of them just high-end users or academics, discuss features and submit patches through our mailing lists. There are no pre-qualifications other than the ability to write good code and useful ideas.

In practice, this means we have a large, diffuse community consisting of a large variety of overlapping groups in which consensus and individual initiative are the motivating factors in governance. As an example, we actually have four different nonprofits, each with their own boards.

Q: How many developers are part of the project and what are the main companies that employ them?

A: Materially, this means that a small number of full-time developers ... about 20 ... write 80% of the code (measured by number of lines) for each new version. The other 20% generally comes in as individual patches from around 180 people each version.

The companies that employ multiple and/or prominent PostgreSQL developers are listed here: `http://www.postgresql.org/about/sponsors`

Q: Why did you join the PostgreSQL project?

A: I'm a professional database geek, and was developing with MS SQL Server. In 1998 I was looking for a SQL-standard, feature-rich database server which would be affordable to my small-business clients, and MSSQL pricing had gone up. I found PostgreSQL and tinkered with it and joined the mailing lists. Shortly thereafter, I found a bug in MSSQL and posted it to a public forum, and the posting was silently removed from the list. In contrast, I found a bug in PostgreSQL, and Tom Lane (later

to be lead programmer) not only immediately responded to my questions, he created a patch for me to test. From them on I was hooked.

Q: How has your involvement in PostgreSQL project affected your professional career?

A: I certainly can't complain. I went from being a small-time small-business tech consultant to a senior engineer at Sun Microsystems whose name is recognized by database geeks around the world.

Q: What is Sun's business models around PostgreSQL?

A: Sun is using PostgreSQL to enhance the value of Solaris, Sun products and Sun hardware through the inclusion of a supported, enterprise-class free DBMS.

Q: How significant is modularization for open source products?

A: That's an essay question, and not one I think I really have an opinion on.

Q: What role does the community play in the development? How is it significant that PostgreSQL is open source? What is the single greatest benefit and biggest challenge from being open source?

A: The community *is* PostgreSQL. So the role it plays in development is doing 100% of it, as well as strategy, goals and promotion.

Our biggest challenge from being open source is competing against other DBMSes with vendor channels, marketing money and a way of business which database users are more familiar with.

Q: What is PostgreSQL primary strategy to get new users: grow the overall SQL market or try to grab users from the competition?

A: I think that more PostgreSQL users are existing database users of some type. We're not really a "starter" database, and tend to attract more professional DBAs than newbies. Often people migrate to PostgreSQL after trying other OS-RDBMSes which are not robust enough, or proprietary systems which are too expensive. Probably even more people use PostgreSQL alongside other DBMSes.

Despite that, though, our license and reputation for security and reliability has caused PostgreSQL to be embedded in hundreds of products that probably would not have used a server-process SQL-RDBMS otherwise.

So, overall, we're not growing the market for databases in general, but we probably *are* growing the market for powerful RDBMS server software.

Q: Are you mainly competing with other open source databases or closed source databases?

A: Mostly closed source databases, I think. We get more migrations from Oracle and Informix than anything else.

Q: What sort of market share does PostgreSQL have in the overall and open source database market have according to your knowledge?

A: Hard to say, because it's hard to define that market. For example, I wouldn't classify the embedded database market as being the same market as server-process SQL-RDBMSes, even though there are products that can be used in both markets (e.g. Firebird). For server-process DBMSes, I don't think that anyone would contest us the #2 spot after MySQL in terms of the number of "users", for some definition of "user".

Q: What are the challenges with community cooperation from the perspective of Sun?

A: Well, it's mostly that the development styles of career corporate programmers and of OSS programmers are very different. So collaboration between the Solaris team and the PostgreSQL community on patches is a bit of a culture clash. Still, we're getting it worked out. Actually, that issue is less pronounced with Sun that it would be with other companies because Sun is an engineering-centric company. I'd imagine that Microsoft staff working in Open Source have far more of a "split personality" issue to deal with.

Still, Sun has a process for building software which is complex and quite well-established within Sun. That process is very different from the PostgreSQL process – which is also several years old. Heck, PostgreSQL uses CVS and Solaris uses a different version control system, that's tricky enough on its own. So I foresee that I'll be troubleshooting coordination issues for as long as I work with Sun.

Q: What is your perception about the free for development and deployment stripped down versions provided all the major proprietary vendors that usually run under the "Express" label? Are these version a reaction to open source or is this simply natural versioning of increasingly complex products? Is the database market being commoditized by open source databases?

A: These are really the same question. The answer is that the "express" versions are the reality of commoditization of the database market. Given that interfaces and specifications for SQL-RDBMSes have been well-defined for a decade, it's only been differentiation of performance and implementation which has allowed expensive proprietary databases to hold on to any market among cost-conscious customers.

What Open Source did was "get the ball rolling" by offering DBMSes which were "good enough" for the bottom 40% of the market (more, now) at a cost of ownership 10x or more lower than the least expensive proprietary database. I believe this process was happening anyway (well-defined market needs always tend towards commoditization), the presence of OSS databases just sped it up substantially.

Q: Do you see potential for cooperation with other open source RDBMS? And if so in what areas? For example several PostgreSQL project members where present at the preliminary meeting to create an Open Source Database Consortium in November 2005.

A: There's a reality and not just a potential, actually. PostgreSQL and MySQL already collaborated on a security patch this spring, and we collaborate on building database tracks for several conferences. We also share ideas all the time (I've discussed performance issues with MySQL and Derby teams, SQLite got a lot of its parser spec from PostgreSQL), and recommend each other's products when appropriate.

So I'd mark it more as "friendly rivalry", at least most of the time. It wasn't always, but in some ways commercial success for some of the OSS-DBMSes made us realize that we have more to worry about from Oracle, Microsoft and IBM than we do from each other.

Q: How significant is standards compliance for the database market in general and PostgreSQL users in specific?

A: Very significant. It's only the SQL standard (and other standards) which has permitted the commoditization process to proceed at all. So I think you'll find that OSS-DBs are now much more concerned with standards than their proprietary counterparts. Or, to be more accurate, OSS-DBMSes try to adhere to the standards whereas IBM, Microsoft and Oracle try to change them.

Q: What, if any, influence does the PostgreSQL project have on the SQL standard process?

A: None.

Q: Is there a need to change this process?

A: Yes, but that goes beyond questions of Open Source. The ANSI-SQL committee, by '96, became captive the most powerful vendors and more concerned with rubber-stamping the features they already had than crafting an interoperable interface, and since '97 have made no attempt to certify compliance at all. Further, ANSI has always suffered from the committee tendency to "never meet a feature they didn't like", causing the SQL standard to become increasingly baroque and unreadable, and pretty much unimplementable. As a result, many vendors and users regard SQL92 as "the real SQL" and later versions as more of a library of exotic features to pick and choose from.

Certainly the SQL standard isn't moving forwards.

Q: Is there anything else you want to say about open source in the relation database market?

A: Sure. We *are* the database market of the future. Just watch us.

### A.5.3 Luke Lonergan

Q: Could you briefly introduce yourself and your role inside the PostgreSQL project?

A: I am co-founder and CTO of a company that commercializes Postgresql.

Q: What are the primary features that set PostgreSQL apart from other RDBMS products?

A: Open source innovation process. Worldwide development community. Shared-nothing parallel implementation (from Greenplum).

Q: Could you give a brief history overview of your companies involvement with PostgreSQL?

A: Starting in 2000, we chose Postgresql for use in our OLTP transaction scaling product, designed to cache transaction workloads in front of Oracle backend systems. We chose Postgresql after evaluating several other choices including instantDB and Cloudscape (Java databases) and MySQL. Postgresql won due to it's more complete support for SQL, Oracle compatible data types, a procedural language that was similar to Oracle (PL/PGSQL) and a strong development community.

Q: Could you explain how the PostgreSQL project works? How is the code developed? How many developers are part of the project and what are the main companies that employ them?

A: Code is developed by a large group of loosely related people, submitted as patches to be reviewed by a smaller more closely related group in the "core", who clean up and apply the code to the main CVS repository. The larger group of contributors is probably 30 in number, the core group might be 10.

There are four main companies that now employ full time Postgresql developers, Greenplum, EnterpriseDB, Command Prompt and Red Hat.

Q: Why did your company decided to join the PostgreSQL project?

A: Two primary reasons: community and leveraged development. We benefit from being part of the community which has a worldwide following. The community helps us to sell our parallel version to their employers who need it, or they are sometimes part of application development companies that adopt our technology to help them sell and support their applications.

The leveraged development allows us to steer the project to deliver features that are important to our customers. We contribute some features, and in turn many developers contribute other parts that improve those features.

I directly employ 3 developers that solely contribute to the open source Postgresql project, and 12 others that focus on the massively parallel version, but we benefit from dozens of other contributors that are now following our lead on features.

Some of the features we have developed and that were adopted and refined by the Postgresql core team include:

- Constraint Exclusion (partitioning)
- Sorting speedup
- COPY speedup
- On-disk Bitmap Index

As a result of our founding of the Bizgres project and the subsequent exchange of both features and ideas, we have been successful in steering Postgres to develop into a leading business intelligence database.

Q: How has your involvement in PostgreSQL project affected your professional career?

A: It has enabled our company to succeed, which has validated my role as technologist. It has also acted as a force multiplier - helping me to achieve my innovation objectives without needing to directly employ a large number of people.

Q: Could you please explain the Greenplum business model? What sort of benefit does PostgreSQL gain from being open source?

A: We have a hybrid closed/open source model. We have a free distribution called Bizgres which is open source and charge a license fee for a closed source version called Bizgres MPP.

We invest our revenue back into development of innovations that our customers need and contribute some of those to Postgres and refine the closed source innovations that are related to the unique parallelism feature.

Q: How significant is modularization for open source products?

A: I don't understand.

Q: What role does the community play in the development? How is it significant that PostgreSQL is open source? What is the single greatest benefit and biggest challenge from being open source?

A: Biggest benefit is the increased connectivity to a wider customer audience through the mailing lists and open discussion enlightened by the access to the source code.

The biggest challenge is the lack of focus of the marketing / product development. The result of the lack of focus has been that the DBMS was not particularly good at any one thing, rather it was being built to be all things to all people. The commercial effort(s) have brought sufficient focus to establish the DBMS as the best technical choice for at least one market - business intelligence.

Q: What are the primary sources of income for Greenplum? Also what is the ratio on the income side between support contracts and licensing. Are there other significant (emerging) revenue streams (certification and books perhaps)?

A: It's all licensing of the closed source enterprise version and recurring support revenue that derives from the licenses. Support revenue derived from high end data warehousing services is a possibility in the future.

Q: Are there any other numbers you can share about Greenplum? Like what percentage is invested into R&D, marketing/sales, community relations.

A: We are 65% R&D, 20% Field services and support and the rest is sales, marketing and G&A.

Q: What is Greenplum primary strategy to get new users: grow the overall SQL market or try to grab users from the competition? Are you mainly competing with other open source databases or closed source databases?

A: We are competing with mainly closed source DBMS - Teradata, Oracle, IBM DB2, MSSQL.

There is a $30B market for Data Warehousing systems and software and we are competing with existing systems along with developing users who previously could not afford data warehousing systems.

Q: What sort of market share does PostgreSQL have in the overall and open source database market have according to your knowledge?

A: I think it's a tiny fraction of the overall market for DBMS, maybe 1-2%. Within open source, maybe it's a 5-7% compared with MySQL.

Q: What are the challenges with community cooperation from the perspective of Greenplum?

A: Adoption of our features is a bit contentious and our efforts seem to generate undue controversy. Our response is to be overly cautious in relations with the community.

Q: What is your perception about the free for development and deployment stripped down versions provided all the major proprietary vendors that usually run under the "Express" label? Are these version a reaction to open source or is this simply natural versioning of increasingly complex products?

A: It's reactionary and designed to FUD the market. I don't think it's working.

Q: Is the database market being commoditized by open source databases?

A: I think the base level of DBMS features are commoditized - it means that the game is now going to be won by continued innovation, rather than just by "being there".

Q: Do you see potential for cooperation with other open source RDBMS? And if so in what areas? For example several PostgreSQL project members where present at the preliminary meeting to create an Open Source Database Consortium in November 2005.

A: Standards are important and where collaborations can path find new innovations, joint work will solidify the leadership of the open source approach.

Q: How significant is standards compliance for the database market in general and Bizgres/PostgreSQL users in specific?

A: Very important for adoption - customers need the security of knowing that they aren't locked into a proprietary approach.

Q: What, if any, influence does the PostgreSQL project have on the SQL standard process? Is there a need to change this process?

A: I am not sure how much impact the community has - I think we can and should play a visible role in the future.

Q: Is there anything else you want to say about open source in the relation database market?

A: Open source is about increased pace of innovation, not just cheaper or free software. We have found that businesses will gladly pay to support a continuing stream of innovation and the open source process is delivering the best value for their money by enabling the OSS community to leverage a shared successful platform.

### A.5.4 Joshua D. Drake

Q: Could you briefly introduce yourself and your role inside the PostgreSQL project?

A: I am Joshua D. Drake (the other Josh). I am a long time contributor to the PostgreSQL project and the current PostgreSQL SPI Liason. I am president of Command Prompt, Inc. the largest dedicated PostgreSQL vendor.

Q: What are the primary features that set PostgreSQL apart from other RDBMS products?

A: The obvious would be we are Open Source. We are also BSD licensed which does not incur the licensing overhead of other known Open Source databases.

From a technical perspective I would say we are the most extensible of the RDBMS products available.

Q: Could you give a brief history overview of your companies involvement with PostgreSQL?

A: We are the oldest PostgreSQL company. We are the largest of the companies that are specializing in PostgreSQL and PostgreSQL only. As a company our involvement spans all the way back to 1997 but we did not become aggressively involved until until past 2000.

We are currently one of the most active companies in the community. Some of our community sponsored items can be found here:

`http://www.commandprompt.com/community`

Q: Could you explain how the PostgreSQL project works? How is the code developed? How many developers are part of the project and what are the main companies that employ them?

A: The first part of this question is probably better answered by someone like Josh Berkus. However, in brief we are a loosely knit band of world wide developers that create the world's most advanced open source database.

The code is developed in an ad-hoc manner with defined style guidelines and review by major contributors.

There are hundreds of developers that contribute to the development of the product. Some only deliver a single patch a year, some several a day.

The major companies that employ PostgreSQL developers are:

- Command Prompt
- EnterpriseDB
- Red Hat

There are also many private employers that help with sponsorship of PostgreSQL developers.

Q: Why did your company decided to join the PostgreSQL project?

A: We noticed a distinct lack of a professional services for PostgreSQL.

Q: How has your involvement in PostgreSQL project affected your professional career?

A: It has allowed me to never have a job again.

Q: Could you please explain the Command Prompt business model? What sort of benefit does PostgreSQL gain from being open source? How does your PostgreSQL services relate to your other RDBMS products and services?

A: Command Prompt is a professional services organization that dedicates itself to consulting, support and training for PostgreSQL. We are also the only company to deliver an enterprise class, integrated replication solution for PostgreSQL.

PostgreSQL gains from being open source by having thousands of eyes at any one given time reviewing, stressing, commenting and developing on its use.

We only support PostgreSQL. We do not support other database products.

Q: How significant is modularization for open source products?

A: I believe it is very important. The more modularized, the more extensible and flexible a product is.

Q: What role does the community play in the development? How is it significant that PostgreSQL is open source? What is the single greatest benefit and biggest challenge from being open source?

A: The community plays a large role in development, as PostgreSQL is primarily developed by the community.

It is significant that a database like PostgreSQL is open source, because we can directly compete with the likes of Oracle and DB2 without the 40k price tag.

The greatest benefit? That entirely depends on who you ask. A CTO is going to tell you total cost of ownership. A geek is going to tell you access to the code. A developer will tell you the ability to fix problems on their own.

However I don't use PostgreSQL because it is OSS. I use and support PostgreSQL because it is the best database available for my customers, regardless of OSS or not.

The biggest challenge would be the continued comparison between us an MySQL. Yes MySQL is good (very good) at certain things but it just can not compete with PostgreSQL on the high end OLTP arena.

Q: What are the primary sources of income for Command Prompt? Also what is the ratio on the income side between support contracts and licensing. Are there other significant (emerging) revenue streams (certification and books perhaps)?

A: The majority of our income comes from services. It is about a 85% services, 15% licensing.

Q: Are there any other numbers you can share about Command Prompt? Like what percentage is invested into R&D, marketing/sales, community relations.

A: I spend approximately 70% of my time between marketing/sales and community relations. The other 30 percent would be broken up between billables and management.

Q: What is Command Prompt primary strategy to get new users: grow the overall SQL market or try to grab users from the competition? Are you mainly competing with other open source databases or closed source databases?

A: Our goal is to make PostgreSQL #1. We will continue to migrate users away from *any* proprietary database.

The majority of work we receive is from people escaping the 40k price tag of other databases.

Q: What sort of market share does PostgreSQL have in the overall and open source database market have according to your knowledge?

A: Excellent question and one nobody can answer definitely. I can tell you that of major installations among high volume databases, the databases that I encounter are: Oracle, PostgreSQL and DB2. In that order.

Q: What are the challenges with community cooperation from the perspective of Command Prompt?

A: I think the biggest challenge is fighting the perception that companies are evil. It took a long time for the community to trust us and there are still some people that don't.

Q: What is your perception about the free for development and deployment stripped down versions provided all the major proprietary vendors that usually run under the "Express" label? Are these version a reaction to open source or is this simply natural versioning of increasingly complex products?

A: They are a reaction to MySQL not PostgreSQL because the express versions can not on any level compete with PostgreSQL. They can however put a serious dent in the marketability of MySQL as that is the MySQL space.

Q: Is the database market being commoditized by open source databases?

A: No. You would need at least another couple of open source databases of enterprise quality to have that happen.

Q: Do you see potential for cooperation with other open source RDBMS? And if so in what areas? For example several PostgreSQL project members where present at the preliminary meeting to create an Open Source Database Consortium in November 2005.

A: I do not see a purpose in that.

Q: How significant is standards compliance for the database market in general and PostgreSQL users in specific?

A: I believe it is very important. As the largest database vendors are generally setting the standard, PostgreSQL following that standard just makes it that much easier for users to migrate.

Q: What, if any, influence does the PostgreSQL project have on the SQL standard process? Is there a need to change this process?

A: None at this time nor do I see a need. The SQL standard process is a clearly defined by large companies.

Q: Is there anything else you want to say about open source in the relation database market?

A: Watch the next 5 years. Large building made of bricks will show that they were created on a foundation of sand.

## A.6 SQLite

### A.6.1 Richard Hipp

Q: Could you briefly introduce yourself and your role inside the SQLite project?

A: I am Richard Hipp. I am the creator, maintainer, and principal author of SQLite.

Q: What are the primary features that set SQLite apart from other RDBMS products?

A: SQLite is serverless. SQL operations are translated into direct disk I/O operations. There is no server.

SQLite uses manifest typing instead of the rigid static typing called for by the SQL standard. Manifest typing is a superset of static typing.

Q: Could you give a brief history overview of SQLite?

A: SQLite version 1 used the GDBM library for disk storage. Version 1 first appeared in 2000. SQLite version 2 appeared in the fall of 2001. Version 2 included a new b-tree backend with ACID transactions to replace GDBM. Version 3 was released in 2004. Version 3 added support for UTF.

Q: Could you explain how the SQLite project works? How is the code developed? How many developers are part of the project and what are the main companies that employ them?

Two developers, myself and Dan Kennedy, have generated almost all of the code for SQLite. There have been some dozen or more other contributors, but their combined contribution is less than 1% of the total code. The last time I measured, my code was 70%, Dan's was 29% and the remaining 1% was the other dozen developers.

I have my own software development company, Hwaci. Over the past 3 years, several companies have contracted with Hwaci to implement enhancements to SQLite. Dan has always been paid for his contributions, through these contracts.

Q: Why did you release SQLite as public domain? Why did you not choose one of the popular open source licenses?

A: SQLite version 1 was GPL. It used the GDBM library which is GPL, so there was no reason to not use the GPL for version 1.

For version 2, I wanted to move to a more liberal license so that people could statically link SQLite with their projects without having to release their own code as GPL. I looked at various BSD-style licenses. But I could not see any advantage to these over just declaring SQLite to be public domain.

Q: What role does the community play in the development? How is it significant that SQLite is public domain? What is the single greatest benefit and biggest challenge from being public domain?

A: The community reports bugs. They are very good at finding bugs that I would have never thought to look for. Other than that, the user community for SQLite has not been very helpful in its development.

The fact that SQLite is in the public domain does not seem to have made any difference regarding user contributions.

Q: How has your involvement in SQLite project affected your professional career?

A: I originally intended to release SQLite into the wild and to never make one dime from it. But things have not worked out that way :-) Over the past three years, I have picked up a lot of new work because of SQLite, so much so that I could, if I wanted to, work on SQLite exclusively.

SQLite has also served as an excellent "resume". I have received numerous, excellent job offers based on my SQLite work (all of which I have declined).

Q: How significant is modularization for open source and public domain products?

A: Modularization is significant to all projects, open source or otherwise. Without good modularization, the complexity of a project rapidly grows to the point that the code is unmanageable.

Q: What are the primary sources of income generated by SQLite? Also what is the ratio on the income side between support contracts and licensing. Are there other significant (emerging) revenue streams (certification and books perhaps)?

A: Most of the money I have made from SQLite comes from companies who have paid me to make enhancements. Most (but not all) of these enhancements have been folded back into the public domain source tree, at the companies request.

Companies also sometimes fly me to their offices to provide a on-site consultation about SQLite.

I also sell the SQLite Encryption Extension and annual maintenance subscriptions. These pay for the website and provide a little extra income but are not huge money makers.

Q: Are there any other numbers you can share about SQLite? Like what percentage is invested into R&D, marketing/sales, community relations.

A: Pretty much 100% of the income from SQLite goes back into R&D. There is no marketing or sales or community relations - unless you count the www.sqlite.org website among those.

Q: What is SQLite primary strategy to get new users: grow the overall SQL market or try to grab users from the competition? Are you mainly competing with other open source databases or closed source databases?

A: I'm not trying to get new users. I provide SQLite to others with the hope that they might find it useful. If they do, then great. If not, my feelings are not hurt. SQLite has turned out to be wildly popular (much to my surprise) but that has nothing to do with any efforts on my part to get new users.

Q: What sort of market share does SQLite have in the overall and open source database market have according to your knowledge?

A: I have no data on the total size of the market and I have precious little data on the number of SQLite users.

The www.sqlite.org website is visited by around 6000 unique IP addresses per day and gets about 1GiB of traffic.

I know that SQLite is used in a lot of widely distributed products (both hardware and software). Some of these products are known. For example, SQLite is used in Apple's OS-X operating system and on Philips MP3 players. Other uses are known to me but I cannot reveal them due to NDAs. I suspect that most uses of SQLite are unknown to me - that it is widely used in many products that I do not know about. For example, GE and Toshiba have both independently contacted me to find out what the US Export Control Number is for SQLite - suggesting that they are using SQLite in some product that is manufactured in the US and shipped overseas. But I have no idea what those products might be. There are three separate Chinese translations of the www.sqlite.org website, suggesting that SQLite is very popular in China, though I have no idea what it might be used for there.

Q: What are the challenges with community cooperation from the perspective of SQLite?

A: In order to keep SQLite in the public domain, I have to have signed copyright release forms from all contributors. I maintain a file of these forms in the firesafe at my office. A sample copyright release can be found at `http://www.sqlite.org/copyright-release.html`

The fact that people are required to sign a legal document in order to contribute to SQLite adds a barrier to entry that is sufficient to discourage many contributors, I suspect.

Q: Aside from your own consulting is there any commercial involvement in the development of SQLite?

A: Many companies have contracted with Hwaci to provide enhancements to SQLite. In a few cases, the companies have publically acknowledged their contribution. For example, AOL publically acknowledged that they contributed some funds toward getting SQLite 3.0.0 release sooner than it would have been otherwise.

Q: What is your perception about the free for development and deployment stripped down versions provided all the major proprietary vendors that usually run under the "Express" label? Are these version a reaction to open source or is this simply natural versioning of increasingly complex products?

A: I'm sorry - I do not really pay attention to these things and am not qualified to say anything useful.

Q: Is the database market being commoditized by open source databases?

A: See the previous answer.

Q: Do you see potential for cooperation with other open source RDBMS? And if so in what areas? For example Zak Grant represented SQLite at the preliminary meeting to create an Open Source Database Consortium in November 2005.

A: I *do* cooperate with other RDBMS projects. I enjoy meeting with developers on other projects. We like to get together at conferences and swap ideas.

Q: How significant is standards compliance for the database market in general and SQLite users in specific?

A: SQL is similar to FORTRAN in the 70s - every vendor uses their own dialect. This is unlikely to change, I think. Each SQL product makes different tradeoffs which end up showing through in the interface.

Q: What, if any, influence does the SQLite project have on the SQL standard process? Is there a need to change this process?

A: I am not aware of any influence by SQLite on the SQL standard.

Q: Is there anything else you want to say about open source in the relation database market?

A: I regret that I know so little about this that there is nothing useful that I could say. :-)

# Appendix B

# Bibliography


ACCESS (2006), 'ACCESS and PalmSource Announce the ACCESS Linux Platform', *ACCESS Co., Ltd* . http://www.access.co.jp/english/press/060214.html [14/08/2006].

Ahmed, Z. (2001), 'Interview with Sleepycat President and CEO, Michael Olson', *winterspeak.com* . http://www.winterspeak.com/columns/102901.html [28/09/2006].

Anonymous (2002), 'bad source code too', *Slashdot* . http://developers.slashdot.org/comments.pl?sid=37305&cid=4011492 [22/08/2006].

ANSI (2006), 'influencing the SQL spec', *ANSI* . http://www.ansi.org/membership/overview/overview.aspx?menuid=2 [13/08/2006].

Apache DB Project (2006), 'Frequently Asked Questions', *Apache DB Project* . http://db.apache.org/derby/faq.html [01/10/2006].

Arnö, K. (2006*a*), 'Code Contributions & Consideration', *Kaj Arnö* . http://www.planetmysql.org/kaj/?p=60 [01/06/2006].

Arnö, K. (2006*b*), 'MySQL Max Build Policy', *Kaj Arnö* . http://www.planetmysql.org/kaj/?p=58 [28/08/2006].

Asay, M. (2006*a*), 'Competing against the void: MySQL and PostgreSQL (Greenplum)', *InfoWorld* . http://weblog.infoworld.com/openresource/archives/2006/07/competing_again.html [26/07/2006].

Asay, M. (2006*b*), 'More on open source's billion-dollar speed limit', *InfoWorld* . http://weblog.infoworld.com/openresource/archives/2006/08/more_on_open_so.html [01/06/2006].

Asay, M. (2006*c*), 'Open source or proprietary: Which is the ideal platform for innovation?', *InfoWorld* . http://weblog.infoworld.com/openresource/archives/2006/08/open_source_or.html [25/08/2006].

Asay, M. (2006*d*), 'Open source stuck south of the $1 billion barrier?', *InfoWorld* . http://weblog.infoworld.com/openresource/archives/2006/08/open_source_stu.html [01/06/2006].

Asay, M. (2006*e*), 'Revising Oracle 10g Express Edition: Is "free $" enough?', *InfoWorld* . http://weblog.infoworld.com/openresource/archives/2006/03/revising_oracle.html [28/09/2006].

Asay, M. N. (2006*f*), Open Souce and the Commodity Urge: Disruptive Models for a Disruptive Development Process, *in* 'Open Sources 2.0', first edn, O'Reilly.

Ascher, D. (2004), 'Dynamic Languages - ready for the next challenges, by design.', *ActiveState* . http://www.activestate.com/Company/NewsRoom/whitepapers_ADL.plex [18/08/2006].

AskMen.com (2006), 'Larry Ellison', *AskMen.com* . http://www.askmen.com/men/may00/24_larry_ellison.html [29/09/2006].

Aslett, M. (2006), 'Ellison says open source needs big business', *Computer Business reviewonline* . http://www.cbronline.com/article_news.asp?guid=4DC42B20-76E7-4C3A-82C2-85B2A9B81238 [11/03/2006].

Babcock, C. (2005), 'Oracle And Microsoft Gain In Database Market', *InfomationWeek* . http://www.informationweek.com/showArticle.jhtml;?articleID=159401656 [06/04/2006].

Baker, M. (2006), The Mozilla Project: Past and Future, *in* 'Open Sources 2.0', first edn, O'Reilly.

Beach, P. (2004), 'The Free Database Niche - Response', *IBPhoenix* . http://www.ibphoenix.com/main.nfs?a=ibphoenix&page=ibp_conputerwoche_response [28/09/06].

Berkus, J. (2005), 'InnoDB and the Compromise of Dual Licensing?', *ITtoolbox blog* . http://blogs.ittoolbox.com/database/soup/archives/innodb-and-the-compromise-of-dual-licensing-6068 [22/08/2006].

Berkus, J. (2006), 'Free DBs: the Final Four', *ITtoolbox blog* . http://blogs.ittoolbox.com/database/soup/archives/007516.asp [03/27/2006].

Bloor, R. (2006), 'Googling Google; Oracle Shudders', *Robin Bloor's blog* . http://www.it-director.com/blogs/Robin_Bloor/2006/6/googling_google_oracle_shudders.html [03/10/2006].

Bradford, R. (2006), 'Testing a new MySQL Transactional Storage Engine', *Ronald Bradford's blog* . http://blog.arabx.com.au/?p=134 [01/09/2006].

Brauch, P. (2000), 'Mythos "Das Web"', *c't magazine* . http://www.heise.de/ct/00/01/003/ [11/06/2006].

Bray, H. (2006), 'US puts Iraqi documents on the Web', *The Boston Globe* . http://www.boston.com/business/articles/2006/03/18/us_puts_iraqi_documents_on_the_web/ [03/29/2006].

Briscoe, B., Odlyzko, A. & Tilly, B. (2006), 'Metcalfe's Law is Wrong', *IEEE Spectrum* . http://spectrum.ieee.org/print/4109 [30/08/2006].

Brooks, F. P. (1995), *The Mythical Man-Month*, anniversary edn, Addison Wesley.

Brunelli, M. (2005), 'SleepyCat CEO: Oracle deal an attempt to disrupt MySQL', *SearchOpenSource.com* . http://searchopensource.techtarget.com/originalContent/0,289142,sid39_gci1140527,00.html [28/09/2006].

Brunelli, M. (2006*a*), 'Oracle overhauls Sleepycat's original Berkeley DB', *SearchOpenSource.com* . http://searchopensource.techtarget.com/originalContent/0,289142,sid39_gci1219254,00.html [19/10/2006].

Brunelli, M. (2006*b*), 'Oracle vs. Microsoft: The open source factor', *SearchOracle.com* . http://searchoracle.techtarget.com/originalContent/0,289142,sid41_gci1195863,00.html [22/08/2006].

Brunelli, M. (2006*c*), 'Sizing up Oracle's open source tactics', *SearchOracle.com* . http://searchoracle.techtarget.com/originalContent/0,289142,sid41_gci1197598,00.html [10/07/2006].

Bruno, M. P. (2001), 'Great Bridge Crumbles In 16 Months', *Find Articles* . http://findarticles.com/p/articles/mi_m0NEW/is_2001_Sept_7/ai_77984287 [26/08/2006].

Cantù, M. (2006), 'DevCo, InterBase, and Firebird', *Marco Cantù's blog* . http://blog.marcocantu.com/blog/Interbase_firebird.html [29/09/2006].

Cantu, C. H. (2006*a*), 'Are you afraid of SomeCo buying Firebird?', *Firebird News* . http://firebirdnews.org/?p=130 [11/03/2006].

Cantu, C. H. (2006*b*), 'DevCo aceitando sugestões', *Firebase* . http://blog.firebase.com.br/?p=43 [29/09/2006].

Cantu, C. H. (2006*c*), 'Exclusive interview with Ann Harrison', *Firebird News* . http://www.firebirdnews.org/?p=151 [23/05/06].

Cantu, C. H. & Cisar, P. (2005), 'Results of Firebird Global Usage Survey (September/2005)', *Firebase* . http://www.firebase.com.br/fb/imgdocs/FB_Survey_2005.pdf [23/05/2006].

Celko, J. (2005*a*), *SQL For Smarties: Advanced SQL Programming*, third edn, Morgan Kaufmann.

Celko, J. (2005*b*), *SQL Programming Style*, Morgan Kaufmann.

Challet, D. & Du, Y. L. (2005), 'Microsopic Model of Software Bug Dynamics: Closed Source versus Open Source'.

Chamas, J. (2005), 'Re: Oracle vs MySql', *MySQL AB Forums* . http://forums.mysql.com/read.php?61,18127,18902#msg-18902 [29/09/2006].

Chelf, B. (2006), 'Measuring software quality'.

Christensen, C. M. (2005), *The Innovator's Dilemma*, first collins business essentials edn, Collins Business Essentials.

Clarke, G. (2006), 'Oracle in talks to consume three open source darlings - report', *http://www.theregister.co.uk* . http://www.theregister.co.uk/2006/02/10/oracle_opens_source_acquisition/ [05/09/2006].

Cone, E. (2003), 'SAP's Silent Database', *Baseline* . http://www.baselinemag.com/article2/0,3959,1089924,00.asp [24/08/2006].

Cowley, S. & Peréz, J. C. (2003), 'Novell prepares to take on Red Hat', *ITworld* . http://www.itworld.com/Tech/2428/031105novellredhat/ [03/27/2006].

creative commons (n.d.), 'Creative Commons Licenses', *creative commons* . http://creativecommons.org/licenses.

Crownston, K. & Howison, J. (2004), 'The social structure of Free and Open Source software development'.

Dargo, D. (2006*a*), 'Insurance Premiums', *Dave Dargo's blog* . http://blogs.ingres.com/davedargo/content/2006-06-08.html [03/07/2006].

Dargo, D. (2006*b*), 'Java DB is now part of Sun's JDK', *Dave Dargo's blog* . http://blogs.ingres.com/davedargo/content/2006-05-04.html [02/09/2006].

Dargo, D. (2006*c*), 'The Broken Covenant of Software Licenses or Don't Stop The Sacrifices', *Dave Dargo's blog* . http://blogs.ingres.com/davedargo/2006/04/04 [03/07/2006].

Darrow, B. (2003), 'Open-Source Database Convergence: MySQL To Adopt SAP DB', *CRN* . http://crn.channelsupersearch.com/news/crn/42184.asp [22/08/2006].

Darrow, B. (2004), 'Gartner: Database Market Stable, With Linux Hot Spot', *CRN* . http://www.crn.com/sections/breakingnews/dailyarchives.jhtml?articleId=18842889 [03/30/2006].

Darrow, B. (2006), 'MySQL Gets Starkey, Buys Netfrastructure', *CRN* . http://www.crn.com/sections/breakingnews/breakingnews.jhtml?articleId=180207118 [16/08/2006].

Database Systems Group (2005), 'PostgreSQL or MySQL?', *Fermi National Accelerator Laboratory* . http://www-css.fnal.gov/dsg/external/freeware/pgsql-vs-mysql.html [27/08/2006].

Debrunner, D. (2006), 'Cloudscape to Derby - Closed to Open Source', *IBM developerWorks blog* . http://www-03.ibm.com/developerworks/blogs/page/djd [01/10/2006].

DiBona, C., Cooper, D. & Stone, M. (2006), *Open Sources 2.0*, first edn, O'Reilly.

Dittmar, D. (2002), 'SAP DB Rebuttal (semi official)', *Slashdot* . http://developers.slashdot.org/comments.pl?sid=37305&cid=4016958 [22/08/2006].

Dougherty, D. (2001), 'LAMP: The Open Source Web Platform', *ONLamp.com* . http://www.onlamp.com/lpt/a/566 [01/06/2006].

Dyck, T. (2002), 'Server Databases Clash', *eWeek.com* . http://www.eweek.com/article2/0,4149,293,00.asp [28/09/2006].

Elephants Dream (2006), 'Elephants Dream', *Elephants Dream* .

Elmasri, R. & Navathe, S. B. (2000), *Fundamentals of Database Systems*, third international edn, Addison-Wesley.

EnterpriseDB (2006*a*), 'About EnterpriseDB', *EnterpriseDB* . http://enterprisedb.com/company/index.do [07/04/2006].

EnterpriseDB (2006*b*), 'About EnterpriseDB', *EnterpriseDB* . http://enterprisedb.com/company/index.do [07/04/2006].

EnterpriseDB (2006*c*), 'EnterpriseDB Announces Its Selection by Sun Microsystems to Provide PostgreSQL Support for Applications Running the Solaris 10 Operating System', *EnterpriseDB* . http://enterprisedb.com/news_events/press_releases/08_08_06.do [14/08/2006].

EnterpriseDB (2006*d*), 'EnterpriseDB Launches Financial Support Program for PostgreSQL Open Source Community at Worldwide PostgreSQL Summit', *EnterpriseDB* . http://enterprisedb.com/news_events/press_releases/06_07_06.do [07/07/2006].

Farr, J. (2006), 'Open letter to the PostgreSQL Community', *Pervasive* . http://www.pervasivepostgres.com/letter.asp [27/08/2006].

Firebird (2005), 'Review of Firebird 2.0', *Firebird Project* . http://www.firebirdsql.org/devel/engine/roadmap2006.html [01/10/2006].

"Fished" (2002), 'Documentation', *Slashdot* . http://developers.slashdot.org/comments.pl?sid=37305&cid=4011538 [22/08/2006].

Fogel, K. (2005), *Producing Open Source Software*, O'Reilly.

Fosdick, H. (2005), 'Capitalizing on Open Source Databases', *dbazine.com* . http://www.dbazine.com/ofinterest/oi-articles/fosdick4 [27/08/2006].

Garret, R. (2006), 'Let's get a real database', *Xooglers* . http://xooglers.blogspot.com/2005/12/lets-get-real-database.html [11/03/2006].

Garry, C. (2005), 'Oracle May Want to Spur MySQL's Growth, not Hurt It', *eWeek.com* . http://www.eweek.com/article2/0,1895,1871112,00.asp [22/03/2006].

Garry, C. (2006), 'IBM Needs to Act Boldly in Database Field', *eWeek.com* . http://www.eweek.com/article2/0,1895,1927234,00.asp [03/27/2006].

Gartner (2006), 'Gartner Says Worldwide Relational Database Market Increased 8 Percent in 2005', *Gartner* . http://www.gartner.com/press_releases/asset_152619_11.html [28/09/2006].

Gilbert, A. (2004), 'Oracle to call on Microsoft in antitrust battle', *cNet News.com* . http://news.com.com/2100-1012-5167927.html?tag=nl [28/09/2006].

Gilbert, A. (2005), 'Larry's war: Oracle vs. SAP', *CNet News.com* . http://news.com.com/Larrys+war+Oracle+vs.+SAP/2100-1014_3-5650929.html [10/05/2006].

Gilfillan, I. (2005), 'Oracle's purchase of InnoDB, their release of Oracle Express, and the effect on MySQL', *Database Journal* . http://www.databasejournal.com/features/mysql/article.php/3561731 [10/05/2006].

Gilfillan, I. (2006), 'MySQL 5.1 - the next generation', *Database Journal* . http://www.databasejournal.com/features/mysql/article.php/3592726 [01/06/2006].

GiST for PostgreSQL (2006), 'Tsearch2 - full text extension for PostgreSQL', *GiST for PostgreSQL* . http://www.sai.msu.su/~megera/postgres/gist/tsearch/V2/ [28/08/2006].

Gomes, L. (2001), 'Is Microsoft secretly using open source?', *ZDnet* . http://news.zdnet.com/2100-9595_22-530081.html [27/07/2006].

Gorman, M. M. (2001), 'IS SQL A REAL STANDARD ANYMORE?', *Whitemarsh Information Systems Corporation* . http://www.wiscorp.com/is_sql_a_real_standard.pdf [06/08/2006].

Great Bridge (2006), 'Great Bridge release on Bruce, Tom, and Jan', *Linuxports* . http://beta.linuxports.com/pgsql-announce/2000-10/msg00002.php [27/08/2006].

Greene, T. C. (2001), 'Ballmer: "Linux is a cancer"', *The Register* . http://www.theregister.co.uk/2001/06/02/ballmer_linux_is_a_cancer [27/07/2006].

Gutmans, A., Bakken, S. & Rethans, D. (2004), *PHP 5 Power Programming*, Prentice Hall.

Hancock, T. (2006), 'Introducing the Open Graphics Project', *Free Software Magazine* . http://www.freesoftwaremagazine.com/node/1665 [28/09/2006].

Herrington, J. (2003), 'The PHP Scalability Myth', *ONJava.com* . http://www.onjava.com/pub/a/onjava/2003/10/15/php_scalability.html [27/08/2006].

Hilf, B. & Asay, M. (2006), 'Bill Hilf interviews Matt Asay at OSCON 2006', *Microsoft Port 25* . http://port25.technet.com/archive/2006/07/31/Bill-Hilf-interviews-Matt-Asay-at-OSCON-2006.aspx [16/08/2006].

Hipp, R. (2006), 'Re: [sqlite] Major projects using SQLite', *SQLite mailing list* . http://www.mail-archive.com/sqlite-users%40sqlite.org/msg16931.html [21/08/2006].

Hoffmeister, J. (2003), 'SAP DB A review on 2 years of Open Source', *SAP DB* . http://www.sapdb.org/7.4/pdf/linuxday2003.pdf [23/08/2006].

Horstmann, J. (2005), Migration to Open Source Databases, Diploma Thesis, Technische Universität. http://cis.cs.tu-berlin.de/Dokumente/Diplomarbeiten/2005/horstmann.pdf.

Horstmann, J. (2006), 'RDBMS Timeline', *OSDB Migration portal.* . http://www.osdbmigration.org/index.php/RDBMS_Timeline [03/29/2006].

hsqldb Development Group (2006), 'HSQLDB', *The hsqldb Development Group* . http://www.hsqldb.org/ [01/10/2006].

Hudson, A. (2006*a*), 'Business Factors in OSS Database Companies', *OSnews* . http://www.osnews.com/story.php?news_id=13823 [23/05/2006].

Hudson, A. (2006*b*), 'Open Source databases rounded up and rodeod', *the inquirer* . http://www.theinquirer.net/default.aspx?article=28201 [11/03/2006].

Hyatt, J. (2006), 'MySQL: Workers in 25 countries with no HQ', *CNNMoney.com* . http://money.cnn.com/2006/05/31/magazines/fortune/mysql_greatteams_fortune/index.htm [01/06/2006].

IBM (2006), 'New to Open source', *IBM* . http://www-128.ibm.com/developerworks/opensource/newto/ [03/27/2006].

IBPhoenix (2003), 'Vulcan Announcement', *IBPhoenix* . http://www.ibphoenix.com/main.nfs?a=ibphoenix&page=vul_announcement [30/09/2006].

ifrOSS (2005), *Die GPL kommentiert und erklärt*, O'Reilly.

Ilan, Y. (2006), 'The Economics of Software Distribution over the Internet Revisited', *first monday* . http://www.firstmonday.org/issues/issue6_12/ilan [23/05/2006].

Ingres (2006*a*), 'Dave Dargo Chief Technology Officer and Senior Vice President of Strategy', *Ingres* . http://www.ingres.com/company/Exec_Dave_Dargo.html [03/07/2006].

Ingres (2006*b*), 'Ingres Introduces Project Icebreaker at LinuxWorld San Francisco', *Ingres* . http://www.ingres.com/news/2006-08-15_Icebreaker.html [16/10/2006].

Jones, P. (2000), 'Brooks' Law and open source: The more the merrier?', *IBM* . http://www-128.ibm.com/developerworks/linux/library/os-merrier.html [30/08/2006].

Kanellos, M. (2005), 'Newsmaker: Gates taking a seat in your den', *News.com* . http://news.com.com/Gates+taking+a+seat+in+your+den/2008-1041_3-5514121-4.html?tag=st.num [27/07/2006].

Kerner, S. M. (2004), 'Latest MySQL Fails to Quiet Licensing Critics', *internetnews.com* . http://www.internetnews.com/dev-news/article.php/3358061 [01/06/2006].

Knowledge@Wharton (2006), 'Oracle trying to cover all its (data) bases', *CNet News.com* . http://news.com.com/Oracle+trying+to+cover+all+its+data+bases/2030-1069_3-6048656.html [06/04/2006].

Krill, P. (2005), 'Ellison pans open source databases', *Infoworld* . http://weblog.infoworld.com/techwatch/archives/004074.html [29/09/2006].

Krishnamurthy, S. (2002), 'Cave or Community? An Empirical Examination of 100 Mature Open Source Projects'.

Kuchinskas, S. (2004), 'Microsoft Ordered to Pull Anti-Linux Ad', *internetnews.com* . http://www.internetnews.com/dev-news/article.php/3400131 [03/27/2006].

Kunze, M. (1998), 'Let There be Light', *heise* . http://www.heise.de/ct/english/98/12/230/ [01/06/2006].

Lacy, S. (2006), 'A Chill in Oracle's Hot Numbers', *BusinessWeek online* . http://www.businessweek.com/technology/content/mar2006/tc20060321_677519.htm [21/03/2006].

Lai, E. (2006*a*), 'Former CEO touts Oracle open-source moves', *LinuxWorld.com.au* . http://www.linuxworld.com.au/index.php/id;469213872;fp;4;fpid;4 [05/08/2006].

Lai, E. (2006*b*), 'Oracle's Sleepycat Purchase Doesn't Rile MySQL Users', *Computerworld* . http://www.computerworld.com/softwaretopics/software/story/0,10801,109018,00.html [28/09/2006].

LaMonica, M. (2005), 'Will open-source crack open database market?', *news.com* . http://news.com.com/2061-10795_3-5716605.html [06/04/2006].

LaMonica, M. (2006*a*), 'IBM to Oracle: you can't buy open source', *CNet News.com* . http://news.com.com/2061-10795_3-6063115.html [20/04/2006].

LaMonica, M. (2006*b*), 'MySQL fills Oracle-consumed hole in database', *cNet News.com* . http://news.com.com/MySQL+fills+Oracle-consumed+hole+in+database/2100-1012_3-6058930.html [22/08/2006].

Laurent, A. M. S. (2004), *Understanding Open Source & Free Software Licensing*, O'Reilly.

Laurie, B. (2006), Open Source and Security, *in* 'Open Sources 2.0', first edn, O'Reilly.

Lavigne, D. (2006), 'OSCON Interviews: Solid', *ITtoolbox blog* . http://blogs.ittoolbox.com/unix/bsd/archives/oscon-interviews-solid-10796 [31/08/2006].

Lehey, G. (2001), 'The Daemon's Advocate: BSD in the News', *Deamon News* . http://ezine.daemonnews.org/200108/dadvocate.html [27/07/2006].

Lemos, R. (2004), 'Linux 'better than proprietary software'', *ZDNet UK* . http://news.zdnet.co.uk/software/linuxunix/0,39020390,39181043,00.htm [01/05/2006].

Lentz, A. (2006), 'Jay Pipes joins MySQL Community dept', *Arjen Lentz's blog* . http://arjen-lentz.livejournal.com/58295.html [01/06/2006].

Lerdorf, R. (2006), 'Getting Rich with PHP', *Rasmus Lerdorf at php|tek 2006* . http://talks.php.net/show/tek06 [11/06/2006].

Lerner, J. & Tirole, J. (2002), 'Some simple economics of open source'.

Lessig, L. (2000), *Code and Other Laws of Cyberspace*, Basic Books.

Loftus, J. (2004), 'Desktop apps ripe turf for open source', *SearchOpenSource.com* . http://searchopensource.techtarget.com/originalContent/0,289142,sid39_gci1011227,00.html [24/08/2006].

Loftus, J. (2005), 'Sun support of PostgreSQL seen as boon to open databases', *SearchOpenSource.com* . http://searchopensource.techtarget.com/originalContent/0,289142,sid39_gci1147541,00.html [27/08/2006].

Lombardi, C. (2006), 'Should Oracle fear open source?', *silicon.com* . http://software.silicon.com/applications/0,39024653,39159164,00.htm [28/05/2006].

Marshall, M. (2006), 'Optaros takes on SpikeSource, other open source start-ups', *SiliconBeat* . http://www.siliconbeat.com/entries/2005/03/10/optaros_takes_on_spikesource_other_open_source_startups.html [05/10/2006].

Marson, I. (2005), 'MySQL and Firebird battle for database top spot', *ZDNet UK* . http://news.zdnet.co.uk/software/applications/0,39020384,39185042,00.htm [23/05/2006].

Martens, C. (2006*a*), 'MySQL sees 'invisible hand' at work in open source', *InfoWorld* . http://www.infoworld.com/article/06/10/02/HNmysqlinterview_1.html [03/10/2006].

Martens, C. (2006*b*), 'SAP to ramp up attempts to woo Oracle user base', *Computerworld* . http://www.computerworld.com.au/index.php/id;144679690;relcomp;1 [22/08/2006].

McClure, S. (1997), 'Object Database vs. Object-Relational Databases', *CA Inc.* . http://www.ca.com/products/jasmine/analyst/idc/14821E.htm [06/04/2006].

McConnachie, D. (2005), 'PostgreSQL 8.1 gets performance boost', *LinuxWorld.com-au* . http://www.linuxworld.com.au/index.php/id;1737879473;fp;4;fpid;3 [03/10/2006].

McConnachie, D. (2006), 'Hilf speaks about Linux through Microsoft eyes', *Computerworld* . http://www.computerworld.com.au/index.php/id;1871876320;fp;2;fpid;4 [03/27/2006].

McCown, S. (2006), 'Inside IBM DB2 Viper', *InfoWorld* . http://www.infoworld.com/article/06/08/14/33FEdb2viper_1.html [24/08/2006].

McPherson, A. (2002), 'Vendor Vision and Strategy', *Computer Associates at the IT Service Management Forum* . http://www.itsmf.com/upload/conference2002/Choosing_Partner.ppt [11/06/2006].

Mickos, M. (2006), 'MySQL CEO Mårten Mickos Answers Your Questions', *Slashdot* . http://interviews.slashdot.org/interviews/06/10/20/1325244.shtml [20/10/2006].

Microsoft (2005), 'Shared Source Licenses', *Microsoft* . http://www.microsoft.com/resources/sharedsource/licensingbasics/sharedsourcelicenses.mspx [03/27/2006].

Montalbano, E. (2006), 'Agassi: MySQL will support SAP this year', *LinuxWorld.com.au* . http://www.linuxworld.com.au/index.php/id;487782658;fp;4;fpid;3 [09/04/2006].

Moore, G. A. (2002), *Crossing the Chasm*, Collins Business Essentials.

Murdock, I. (2006), Open Source and the Commoditization of Software, *in* 'Open Sources 2.0', first edn, O'Reilly.

MySQL AB (2003), 'MaxDB by MySQL Now Available', *MySQL AB* . http://www.mysql.com/news-and-events/press-release/release_2003_35.html [24/08/2006].

MySQL AB (2006*a*), 'Changes in release 3.23.15 (08 May 2000)', *MySQL AB* . http://dev.mysql.com/doc/refman/4.1/en/news-3-23-15.html [01/06/2006].

MySQL AB (2006*b*), 'Changes in release 3.23.34a (11 March 2001)', *MySQL AB* . http://dev.mysql.com/doc/refman/4.1/en/news-3-23-34a.html [01/06/2006].

MySQL AB (2006*c*), 'Changes in release 4.1.7 (23 October 2004: Production)', *MySQL AB* . http://dev.mysql.com/doc/refman/4.1/en/news-4-1-7.html [01/06/2006].

MySQL AB (2006*d*), 'Changes in release 5.0.15 (19 October 2005: Production)', *MySQL AB* . http://dev.mysql.com/doc/refman/5.0/en/news-5-0-15.html [01/06/2006].

MySQL AB (2006*e*), 'My Forge', *MySQL AB* .

MySQL AB (2006*f*), 'MySQL Contributor License Agreement', *MySQL AB* . http://forge.mysql.com/wiki/MySQL_Contribution_License_Agreement [01/06/2006].

NAP (1999), 'The Rise of Relational Databases', *National Academy Press* . http://www.nap.edu/readingroom/books/far/ch6.html [04/13/2006].

Net Applications (2006), 'Browser Market Share for Calendar Q2, 2006', *Market Share by Net Applications* . http://marketshare.hitslink.com/report.aspx?qprid=0&qpmr=15&qpdt=1&qpct=3&qptimeframe=Q&qpsp=29 [28/09/2006].

Olson, M. (2006), Dual Licensing, *in* 'Open Sources 2.0', first edn, O'Reilly.

OneStat.com (2006), 'Global usage share Mozilla Firefox has increased according to OneStat.com', *OneStat.com* . http://www.onestat.com/html/aboutus_pressbox44-mozilla-firefox-has-slightly-increased.html [28/09/2006].

Onetti, A. & Capobianco, F. (2005), Open Source and Business Model Innovation. The Funambol case, *in* 'First International Conference on Open Source Systems'. http://www.funambol.com/articles/OSS05_paper.pdf [10/09/2006].

Oracle (2006), 'Oracle Reports Q3', *Oracle* . http://www.oracle.com/corporate/investor_relations/3q06-pressrelease-march.pdf [01/05/2006].

Oram, A. (2004), 'Why MySQL grew so fast (news from the 2004 MySQL Users Conference)', *ONLamp.com* . http://www.oreillynet.com/onlamp/blog/2004/04/why_mysql_grew_so_fast_news_fr.html [01/06/2006].

O'Reilly, T. (2004), 'Open Source Paradigm Shift', *O'Reilly* . http://tim.oreilly.com/articles/paradigmshift_0504.html [03/27/2006].

O'Reilly, T. (2005), 'What Is Web 2.0', *ONLamp.com* . http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html [17/08/2006].

Orsini, F. (2006), 'Java DB is now part of Sun's JDK', *Francois Orsini's blog* . http://weblogs.java.net/blog/forsini/archive/2006/06/java_db_is_now.html [01/10/2006].

OSDB Consortium (2005), 'About Hackers', *Open source database consortium* . http://lists.osdbconsortium.org/mailman/listinfo/hackers [18/08/2006].

OSDB Consortium (2006), 'influencing the SQL spec', *Open Source Database Consortium mailinglist* . http://lists.osdbconsortium.org/pipermail/hackers/2006-July/000121.html.

OSI (2006*a*), 'The Approved Licenses', *Open Source Initiative* . http://opensource.org/licenses/ [03/29/2006].

OSI (2006*b*), 'The Open Source Definition', *Open Source Initiative* . http://opensource.org/docs/definition.php [04/04/2006].

OTEC (2006), 'U.S. Computer Software Industry: Size, Firms, Establishments, Employments, Receipts', *Office of Technology and Electronic Commerce* . http://web.ita.doc.gov/ITI%5CitiHome.nsf/AutonomyView/87200518f179196c85256cc40077ede1 [01/05/2006].

Pavlicek, R. (2004), 'MySQL's MaxDB: A work in progress', *Newsforge* . http://programming.newsforge.com/article.pl?sid=04/01/26/187210&tid=72&tid=140&tid=25&tid=26&tid=31 [22/08/2006].

Perens, B. (2005), 'The Emerging Economic Paradigm of Open Source', *Bruce Perens* . http://www.perens.com/Articles/Economic.html [11/04/2006].

PostgreSQL (2006*a*), 'Appendix E. Release Notes', *PostgreSQL Project* . http://www.postgresql.org/docs/current/static/release.html [16/08/2006].

PostgreSQL (2006*b*), 'Developers', *PostgreSQL Project* . http://www.postgresql.org/developer/bios [27/08/2006].

PostgreSQL (2006*c*), 'History', *PostgreSQL Project* . http://www.postgresql.org/about/history [26/08/2006].

PostgreSQL (2006*d*), 'Sponsors', *PostgreSQL Project* . http://www.postgresql.org/about/sponsors [27/08/2006].

Poulsen, K. (2001), 'Borland Interbase backdoor exposed', *The Register* . http://www.theregister.co.uk/2001/01/12/borland_interbase_backdoor_exposed/ [30/09/06].

Powell, J. & Moris, F. (2002), 'Different Timelines for Different Technologies: Evidence from the Advanced Technology Program', *Advanced Technology Program* . http://www.atp.nist.gov/eao/ir-6917/ir-6917.pdf [06/04/2006].

Raymond, E. S. (2001), *The Cathedral and the Bazaar*, revised edn, O'Reilly.

Raymond, E. S. (2006), 'The Halloween Documents', *Eric S. Raymond's Home Page* . http://www.catb.org/~esr/halloween/ [28/09/2006].

Reavis, J. (2000), 'Linux vs. Microsoft: Who Solves Security Problems Faster?', *CSOinformer* . http://csoinformer.com/research/solve.shtml [27/07/2006].

Red Herring (2006*a*), 'IBM Profits Reflect Turnaround', *Red Herring* . http://www.redherring.com/Article.aspx?a=16545&hed=IBM+Profits+Reflect+Turnaround [28/09/2006].

Red Herring (2006*b*), 'MySQL Opens Up Its Database', *Red Herring* . http://www.redherring.com/Article.aspx?a=16651&hed=MySQL+Opens+Up+Its+Database [02/05/2006].

Refractions Research (2006), 'What is PostGIS?', *PostGIS* . http://postgis.refractions.net/ [28/08/2006].

Riding, A. (2005), 'France Detects a Cultural Threat in Google', *New York Times* . http://www.nytimes.com/2005/04/11/technology/11google.html?ex=1270872000&en=6d6222e3d1eab2f2&ei=5090 [04/08/2006].

robertb@sraapowergres.com (2005), 'SRA America allies with Afilias to develop SLONY Support', *PostgreSQL Project* . http://www.postgresql.org/about/news.380 [27/08/2006].

Ruizendaal, P. (2005), Moving Oracle applications to Firebird, *in* 'Firebird Conference 2005'. http://www.ibphoenix.com/downloads/FirebirdConf2005/TOOLS-A303-R3/TOOLS-A303-R3.zip [19/10/2006].

Salkever, A. (2003), 'A Baby Database's Chance to Grow Up?', *Benchmark Capitel* . http://www.benchmark.com/news/sv/2003/06_04_2003.php [22/08/2006].

Sanders, T. (2005), 'SAP dismisses open source innovation', *vnunet* . http://www.vnunet.com/vnunet/news/2145809/sap-dismisses-open-source [27/07/2006].

Sanders, T. (2006), 'Microsoft Takes On SAP, Oracle With New ERP Suite', *CRMBuyer* . http://www.crmbuyer.com/story/51164.html [24/08/2006].

SAP (2002), 'SAP DB - The Open Source Database from SAP AG', *SAP DB* . http://www.sapdb.org/pdf/backsapdb_eng.pdf [23/08/2006].

SAP (2006), 'SAP DB under GNU GPL', *SAP AG* . http://www.sapdb.org/sap_db_gnu.htm [03/27/2006].

Schindler, E. (2006), 'What Gartner Is Telling Your Boss', *DevSource* . http://www.devsource.com/article2/0,1895,2020837,00.asp [28/09/2006].

SEC (2004), 'Microsoft Form 10-Q for September 2004', *Shareholder.com* . http://microsoft.shareholder.com/redesign/EdgarDetail.asp?CIK=789019&FID=1193125-04-189841&SID=04-00 [01/05/2006].

Shankland, S. (2001), 'Red Hat sought help in database plan', *cnet News.com* . http://news.com.com/Red+Hat+sought+help+in+database+plan/2100-1001_3-268915.html [27/08/2006].

Shankland, S. (2004), 'MySQL addresses open-source license problem', *ZDNet* . http://news.zdnet.com/2100-3513_22-5173014.html [21/08/2006].

Shankland, S. (2006), 'Oracle tried to buy open-source MySQL', *News.com* . http://www.news.com/Oracle+tried+to+buy+open-source+MySQL/2100-7344_3-6040197 [11/03/2006].

Shankland, S. & LaMonica, M. (2004), 'IBM to make Java database open source', *c|net news.com* . http://news.com.com/IBM+to+make+Java+database+open+source/2100-7344_3-5291025.html [03/29/2006].

Shapiro, C. & Varian, H. R. (1999), *Information Rules*, Harvard Business School Press.

Simeonov, S. (2006), 'Metcalfe's Law: more misunderstood than wrong?', *Simeon Simeonov's blog* . http://simeons.wordpress.com/2006/07/26/metcalfes-law-more-misunderstood-than-wrong [30/08/2006].

Siteforum (2003), 'SAP, MySQL and city of Munich - good news for SITEFORUM customers', *Siteforum* . www.siteforum.com/cms/news_-_press/sap__mysql_and_city_of_munich_-_good_news_for_siteforum_customers_eng.html [24/08/2006].

Sleepycat (2006), 'Licensing', *Sleepycat* . http://www.sleepycat.com/company/licensing.html [28/09/2006].

Smith, S. (2006), 'EnterpriseDB - WHERE IS THE SOURCE????', *Stewart Smit's blog* . http://www.flamingspork.com/blog/2006/02/16/enterprisedb-where-is-the-source/ [03/10/2006].

Songini, M. (2003), 'SAP inks open-source database deal', *Computerworld* . http://www.computerworld.com/databasetopics/data/software/story/0,10801,81603,00.html [10/05/2006].

SQLite (2004), 'SQLite Copyright', *SQLite* . http://www.sqlite.org/copyright.html [03/29/2006].

SQLite (2006*a*), 'Appropriate Uses For SQLite', *SQLite* . http://sqlite.org/whentouse.html [18/08/2006].

SQLite (2006*b*), 'Database Speed Comparison', *SQLite* . http://sqlite.org/speed.html [18/08/2006].

SQLite (2006*c*), 'SQL Features That SQLite Does Not Implement', *SQLite* . http://sqlite.org/omitted.html [18/08/2006].

SQLite (2006*d*), 'SQLite', *SQLite* .

SQLite (2006*e*), 'SQLite Copyright', *SQLite* . http://www.sqlite.org/cvstrac/wiki?p=SqliteUsers [18/08/2006].

SQLite (2006*f*), 'SQLite Copyright', *SQLite* . http://sqlite.org/compile.html [18/08/2006].

SSC (2006), 'Guide to Legal Issues in Using Open Source Software v2', *State Services Commission* . http://www.e.govt.nz/policy/open-source/open-source-legal2/legal-issues-v2.pdf [01/05/2006].

Stallman, R. (2001), 'Free Software: Freedom and Cooperation', *GNU Project* . http://www.gnu.org/events/rms-nyu-2001-transcript.txt [23/05/2006].

Stallman, R. (2005), 'The Free Software Definition', *Free Software Foundation* . http://www.gnu.org/philosophy/free-sw.html [03/27/2006].

Stellman, A. & Greene, J. (2006), 'What Corporate Projects Should Learn from Open Source', *O'Reilly OnLamp* . http://www.onlamp.com/lpt/a/6486 [07/07/2006].

Stoll, B. L. (2006), Spass und Software-Entwicklung Zur Motivation von Open-Source-Programmierern, Diploma Thesis, Universität Zürich. http://www.dissertationen.unizh.ch/2006/luthigerstoll/diss.pdf.

Sun (2006), 'Sun Microsystems Launches OpenSPARC Project - Ignites New Open Source Community for Breakthrough UltraSPARC T1 Processor', *Sun Microsystems* . http://www.sun.com/smi/Press/sunflash/2005-12/sunflash.20051206.4.xml [03/29/2006].

Taft, D. K. (2006*a*), 'Apple's Leopard Hails Ruby on Rails', *eWeek* . http://www.eweek.com/article2/0,1759,2000971,00.asp [18/08/2006].

Taft, D. K. (2006*b*), 'Goldman Sachs Trades Up and Scales Out', *eWeek* . http://www.eweek.com/article2/0,1759,1959552,00.asp [24/08/2006].

Telegraph.co.uk (2006), 'Open-source, software-as-a-service and grid computing are all major developments still to be tackled', *Telegraph.co.uk* . http://www.telegraph.co.uk/money/main.jhtml?xml=/money/2006/07/03/ccorac103.xml&sSheet=/money/2006/07/03/ixcoms.html [03/09/2006].

Tenney, D. (2001), 'Bruce Momjian on PostgreSQL, Great Bridge, the Future and the Past', *LWN.net* . http://lwn.net/2001/features/Momjian/index.php3 [26/08/2006].

Tomasi, C. & Steppe, K. (2006), 'Interview: Kevin Rose', *ChuckChat* . http://www.chuckchat.com/technorama/?p=149 [27/08/2006].

Treat, R. (2006*a*), 'What the free non-free databases signal', *Robert Treat's blog* . http://people.planetpostgresql.org/xzilla/index.php?/archives/142-What-the-free-non-free-databases-signal.html [11/03/2006].

Treat, R. (2006*b*), 'What to do with 20 million dollars', *Robert Treat's blog* . http://people.planetpostgresql.org/xzilla/index.php?/archives/251-What-to-do-with-20-million-dollars.html [10/08/2006].

Tridgell, A. (2000), 'Samba-TNG fork', *Samba* . http://www.samba.org/samba/tng.html [27/07/2006].

Updegrove, A. (2006), 'Where (if anywhere) are the Boundaries of the Open Source Concept?', *Sun* . http://www.consortiuminfo.org/standardsblog/article.php?story=20060324074006276 [03/29/2006].

Urlocker, Z. (2006*a*), 'A Billion Page Views Per Day', *TheOpenForce.com* . http://www.theopenforce.com/2006/08/a_billion_page_.html [17/08/2006].

Urlocker, Z. (2006*b*), 'Gartner on Open Source Databases', *Zack Urlocker's blog* . http://www.theopenforce.com/2006/09/gartner_on_open.html [28/09/2006].

Urlocker, Z. (2006*c*), 'Oracle FAQ Runs MySQL', *TheOpenForce.com* . http://www.theopenforce.com/2006/03/oracle_faq_runs.html [01/06/2006].

Urlocker, Z. (2006*d*), 'Should Oracle fear open source?', *TheOpenForce.com* . http://www.theopenforce.com/2006/05/should_oracle_f.html [28/05/2006].

Vaas, L. (2004), 'CA Open-Sources the Heart of Its Ingres Database', *eWeek* . http://www.eweek.com/article2/0,1895,1599395,00.asp [03/27/2006].

Vaas, L. (2005*a*), 'AJAX, Java DB a High-Octane Match', *eWeek* . http://www.eweek.com/article2/0,1759,1902407,00.asp [18/08/2006].

Vaas, L. (2005*b*), 'MySQL 5.0 Stars at Users Conference', *eWeek.com* . http://www.eweek.com/article2/0,1759,1788027,00.asp [16/08/2006].

Vaas, L. (2006*a*), 'Enterprise Users Calm as MySQL Community Freaks', *eWeek.com* . http://www.eweek.com/article2/0,1895,1930346,00.asp [23/05/2006].

Vaas, L. (2006*b*), 'MySQL Scores the Brains Behind Firebird', *eWeek.com* . [28/09/2006].

Vaas, L. (2006*c*), 'Oracle Finds the Flaw in MySQL's Business Plan', *eWeek.com* . http://www.eweek.com/article2/0,1895,1869989,00.asp [23/05/2006].

Valderrama, C. (2005), 'Interbase roadmap.', *Claudio Valderrama's blog* . http://www.cvalde.net/ibRoadmap.htm [29/09/2006].

Waldman, S. (2006), 'Who knows?', *The Guardian* . http://technology.guardian.co.uk/online/news/0,12597,1335892,00.html [03/29/2006].

Walli, S. (2006), 'Shai Agassi, SAP, and Open Source Software', *Benchmark Capitel* . http://stephesblog.blogs.com/my_weblog/2006/01/shai_agassi_sap.html [24/08/2006].

Wasserman, T. J. (2006), 'Leverage your PostgreSQL V8.1 skills to learn DB2, Version 8.2', *IBM* . http://www-128.ibm.com/developerworks/db2/library/techarticle/dm-0603wasserman2/ [10/09/2006].

Waters, R. (2006*a*), 'Oracle considers venturing into Linux', *Financial Times* . http://www.ft.com/cms/s/7354696c-cd86-11da-afcd-0000779e2340.html [17/04/2006].

Waters, R. (2006*b*), 'Transcript: FT interview with Larry Ellison', *Financial Times* . http://www.ft.com/cms/s/5f7bdc18-ce85-11da-a032-0000779e2340.html [03/09/2006].

Weber, S. (2004), *The Success of Open Source*, Harvard University Press.

Weiss, T. (2006), 'MySQL acquiring data management system vendor Alzato', *Computerworld* . http://www.computerworld.com/databasetopics/data/software/story/0,10801,86048,00.html?from=imuheads [03/04/2006].

Weiss, T. R. (2003), 'MySQL acquiring data management system vendor Alzato', *Computerworld* . http://www.computerworld.com/databasetopics/data/software/story/0,10801,86048,00.html?from=imuheads [04/04/2006].

Whiting, R. (2003), 'Sybase Strives For Relevance', *InformationWeek* . http://www.informationweek.com/news/showArticle.jhtml?articleID=15202018 [24/08/2006].

Williams, M. (2006), 'Open source needs big vendors to thrive: Ellison', *Computerworld* . http://www.computerworld.co.nz/news.nsf/devt/D0B480B1C0BF93C5CC25712C00744A09 [03/29/2006].

Williams, S. (2002), *Free as in Freedom*, O'Reilly.

Wolf, U. (2000), '0:1 für Unterschleißheim oder Wolperdinger auf LSB', *Linux Magazin* . http://www.linux-magazin.de/Artikel/ausgabe/2000/12/Login/Login.html [27/07/2006].

Woods, D. & Guliani, G. (2005), *Open Source for the Enterprise*, first edn, O'Reilly.

Worsley, J. & Drake, J. (2001), *Practical PostgreSQL*, O'Reilly. http://www.faqs.org/docs/ppbook/x1277.htm [24/08/2006].

Yuhanna, N. (2006), 'Forrester Wave On Open Source Databases', *Forrester* . `http://mysql.com/news-and-events/on-demand-webinars/Forrester_Sept_13.pdf` [19/10/2006].

Zaitsev, P. (2006), 'Leaving MySQL, MySQL Consulting', *MySQL Performance blog* . `http://www.mysqlperformanceblog.com/2006/07/31/leaving-mysql-mysql-consulting/` [01/06/2006].