Open Source Software Licensing and Business Models

Lutfus Sayeed, Sameer Verma, Ahmet Bora Aslan

College of Business, San Francisco State University (lsayeed@sfsu.edu)

Abstract

The primary goal of this research is to study the role of licensing in the software industry. First, the present research will compare proprietary and Open Source Software (OSS) licensing. Then, the study will compare existing business models associated with various open source software licensing approaches. Open source software is in early stages of its life cycle and since the source code is open or freely distributable, licensing will play a big role in the success of projects that use open source software. Therefore, the results of the present paper would shed light on licensing issues that should be considered carefully when developing an IT application using open source software.

1. Open Source Software Licensing and Business Models

Software developers market their products based on a licensing method. Currently, two major licensing methods dominate software licensing- proprietary and open source. The licensing methods influence the business models used by the software developers to market their products. Therefore, business models and software licensing methods are interlinked. Further, software developers use a multiplicity of licensing methods within the open source approach.

The primary goal of this research is to study the role of licensing in the software industry. First, the present research will compare proprietary and Open Source Software (OSS) licensing. Then, the study will compare existing business models associated with various open source software licensing approaches.

Open source software is in early stages of its life cycle and since the source code is open or freely distributable, licensing will play a big role in the success of projects that use open source software. Therefore, the results of the present paper would shed light on licensing issues that should be considered carefully when developing an IT application using open source software.

2. Characteristics of Proprietary Versus Open Source Software

Proprietary software, including the code used to create it, is protected by a patent or trademark. The patent is not free of distribution restrictions, and is rarely free of cost. Proprietary software is also defined as the software whose source code is secret and belongs to a specific individual or a company. Proprietary software has following characteristics:

- Source code is hidden and only available as binaries to end-users.
- Ownership of the software belongs to a specific company, group or individual.
- Proprietary software cannot be redistributed, modified, or sold without the permission of the owner.
- The owner company decides on the release date of the proprietary software.
- Guarded source code software is the basis of the revenue stream.

In contrast, open source software is developed by freelancing but collaborating programmers using freely distributed source code and the communication facilities of the Internet [7]. Open source software projects have the following common criteria:

- The program should include source code and should allow distribution in source code as well as compiled form. When some form of a product is not distributed with source code there should be a well-publicized means of obtaining the source code for no more than a reasonable reproduction cost, preferably, downloading via the internet without charge.
- The open source software should allow modifications and derived works, to be re- distributed under the same terms as the license of the original software. The source code form of the open source software is the preferred form in which a programmer would modify the program. Intermediate forms such as the output of a preprocessor or translator are not allowed. The rights attached to the program apply to all to whom the program is redistributed without the need for execution of an additional license by those parties
- The open source license does not restrict any part from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources.
- The rights attached to the program must not depend on the program's being part of a particular software distribution. If the program is extracted from that distribution and used or distributed within the terms of the program's license, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the original software distribution.

3. Licensing

Companies that practice the traditional software model use closed source software licenses. The closed source license is mainly in the form of the end user license agreement and is distributed with the software. The end user license agreement assures that any software given to a customer can't be copied, redistributed or modified without the permission of the company. Since the commercial software products are in the form of binary and not source code, software products can be licensed per-user, per-machine, per- CPU, per concurrent-user or for an entire organization under closed source software licensing. Closed source license may require a royalty fee and in some versions it has an expiration date which the customer should purchase the software again or buy the new versions of the product.

In contrast, the nature of the open source software is that no right to use license fee or right to re-distribute license fee is charged from the users. There is no royalty charged for neither businesses nor individuals for selling the "source code" of any open source software. However open source is not public domain. When the developer releases the source code to the public under the terms of an open source license, he or she does not relinquish the copyright or patent rights. Open source licenses grant conditional rights to members of the general public willing to comply with the terms of the open source license. Developers can still sell private licenses to individuals and companies that are unwilling or unable to comply with the terms of the open source license.

There are many licenses written for the open source projects but only five to six licenses are widely used. These include Apache Software License, Apple Public Source License, Artistic License, BSD License, GPL License, IBM Common Public License, IBM Public License, Intel Open Source License, Jabber Open Source License, and MIT License. In reality, GPL licensing accounts for 82% of the open source projects [3].

The present paper will discuss several business models based on the licensing approaches within the open source movement.

4. Open Source Licensing

Before we discuss the open source licensing fundamentals, it is essential that we define several terms that are used in the open source community. The Free Software Foundation (FSF) has determined some of these definitions that are widely used in the open source community [9]. FSF's definition of "free" software can be used interchangeably with "open source" software coined by Eric Raymond [7]. FSF's terms and licenses are widely accepted in the open source community. One of the terms frequently used by the Open Source Software community is "Copyleft." Copyleft is a general method for making free software and requiring all modified and extended versions of the program to be free software [13]. Copyleft is reversed from the name Copyright to show that open source software is not taking away the user's freedom. Copyleft suggests that anyone who distributes the software, with or without changes, must pass along the freedom to further copy and change it.

By definition, open source software licenses "may not require a royalty or other fee for such sale" [9]. Open source has the effect of setting the license price to zero. By doing so, a copy of the software given to any user may be turned around and redistributed to others without charging them a license fee. The rationale behind this movement is best described by [9] as "arrangements to make people pay for using a program, including license of copies, always incur a tremendous cost to society through the cumbersome mechanism necessary to figure out how much a person must pay….. Extracting money from users of a program by restricting their use of it is destructive because the restrictions reduce the amount of wealth that humanity derives from the program." The nature of the open source software is that no right to use license fee or right to re-distribute license fee is charged from the users. There is no royalty charged for either businesses or individuals for selling the "source code" of any open source software.

However, open source is not public domain software [11]. That is, when the developer releases the source code to the public under the terms of an open source license, he or she does not relinquish the copyright or patent rights [10]. Open source licenses grant conditional rights to members of the general public willing to comply with the terms of the open source license. Developers can still sell private licenses to individuals and companies that are unwilling or unable to comply with the terms of the open source license. Further, open source licensing must not place restrictions on other software that is distributed along with the licensed software. For example, the license must not insist that all other programs distributed on the same medium must be open-source software. The rationale behind this idea is that distributors of open-source software have the right to make their own choices about their own software. The license must allow modifications and derived works, and distribution under the same terms as the original software license.

One of the protections that the developers are granted in an open source license as the copyright holder of an original work is that they are free to provide their work to others under a different, private license. In this way, they are giving something important to the community by releasing the source, while not sacrificing their right to earn money by selling private licenses. Also, as copyright holders, nothing keeps them from creating new proprietary products based on the original work.

Over the years there have been licenses written for use with open-source software, so that companies can match their particular situation. Companies or individuals can write their new licenses either by creating it from scratch or by modifying an existing license. Tim O'Reilly [6], one of the pioneers of the open source community, described the open source licensing as " the licenses that fosters cooperation, sharing and symbiosis between software creators and software consumers."

Open Source Software has a zero price value or no right to use licensee fee. However, this does not mean that open source software has no software licenses associated with it. In fact all of the open source software require more license agreements because license sets the rule of the game. As we stated above the rationale behind the open source

movement is that the community should benefit from the open source [2]. Thus, there should be licenses that are designed for utilitarian reasons, in order to promote the use, distribution, and creation of useful software. These licenses not only serve individuals but also help companies to build their business models when combining their business models with the open source software.
There are many licenses written for the open source projects but only five to six licenses are widely used. Figure 1 shows the distribution of the licensing among 25,000 open source projects that we gathered from www.fsf.org.

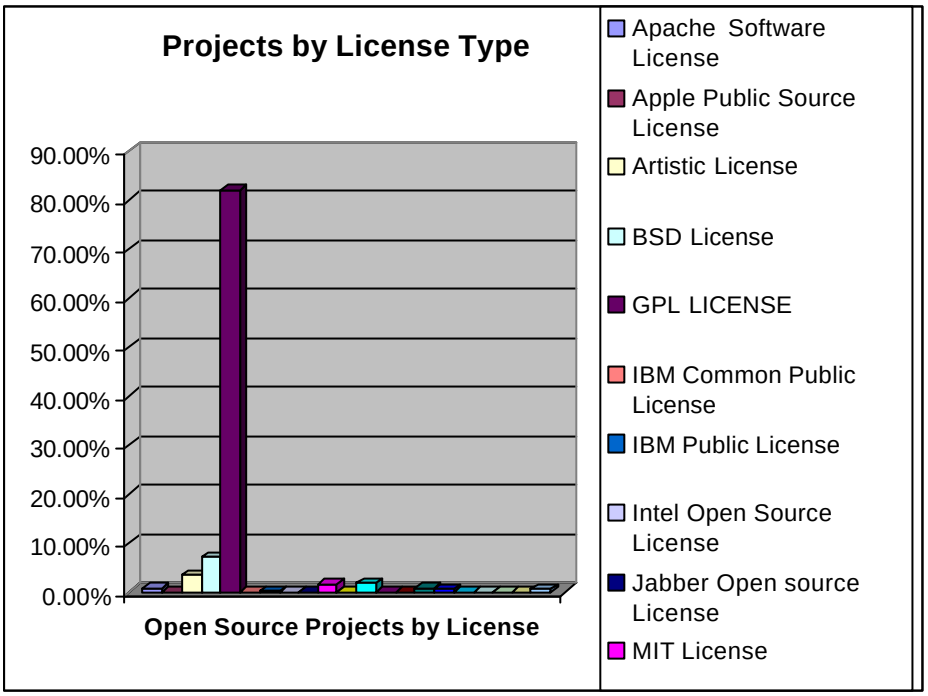

Figure 1. Open Source Projects by License Type
As seen from above, GPL license dominates the open source projects with 82.23%.

5. Business Models

Traditional Business Model

Traditionally, software is considered a product. The software companies provide all of the value to customers and they realize revenues and profits in return for that value through traditional software license fees. The traditional model develops the software in terms of the business rules. The revenues come from selling the software. There are expenses such as engineering costs, marketing costs and other administration costs. US software companies operate under SEC rules and they develop the software according to its marketability [1]. In other words, the software companies do not develop any software product that would have no market value. Companies would like to market new versions of their product when the product has market value not when the product is absolutely fixed from the bugs. In the traditional embedded system development model, there are two choices: build, or buy. That is, expend internal resources to develop the required software, or license a ready-made system and pay the provider the required fees. Either way, there is a cost to the customer. Traditionally, the "build" scenario means hiring a team of experts to address all the technology and software development requirements. The breadth and depth of that team depends on many factors, including the nature and complexity of the required technologies and software, the project's time-to-market requirements, and the willingness of management to expose the program to risk. Similarly, in the "buy" scenario, when a proprietary system is acquired from a vendor, the vendor must absorb these same costs -- and so they are funded by a combination of purchase price, support fees, royalties, and professional services [4].
Traditional business models use closed source licensing where the customers have no right to modify or customize the software. The companies that practice traditional software business model are Microsoft, Oracle, Siebel, and Adobe.

6. Open Source Business Model

How can open source, a model of doing business that flies in the face of age-old business logic, coexist with traditional, closed, models of intellectual property licensing?

The key word in understanding how open source companies generate money is ancillary services [5]. The irony behind open source business model is that they actually make no money on the actual product they invented, but rather, generate revenues from ancillary services like systems integration, support, tutorials and documentation. Open source companies have successfully leveraged their products and services from the web and have shifted the commercial value away from the tangible product they are offering to clusters of services that facilitate, enhance, and integrate the actual product. This is demonstrated by companies such as Red Hat (LINUX), IBM (Apache), Netscape, a host of web-specific technologies such as Java or Perl, and a host of web-specific technology companies such as

Sendmail. This commercial shift emanates from the fact that the real value of the software is negligible if the ancillary services are not offered with it. Imagine using a freeware without knowing 90% of its potential capability. However it is not simply the revenue streams generated by the ancillary services that make open source a compelling business model.

Open source also cuts down on essential research and development costs while at the same time speeding up delivery of new products. This ironic situation stems from the collective free enhancement that an open source community offers. Members themselves provide free research and development by contributing new solutions, features, and ideas back to the open source project as a whole. The open source company then simply dictates what enhancements it will include in its next version and reap the rewards of the work of thousands of highly skilled developers without paying them anything. Open source business models have one of the strongest marketing strategies around: viral marketing itself [8]. Because open source products are typically released for free, open source companies that can produce quality products and generate a good reputation can almost immediately grab huge shares of any market based on the complex and far-reaching global referral networks generated by these savvy users.
Commercial software companies face many challenges in growing their business in today's fast economy and competitive industry environment. Some of these challenges continue to create new products and bring in new incremental revenue, improve new product quality at first release, sustaining engineering support in current and older version of the products while driving innovations in new releases. Recently, many companies proposed the use of an open source development model as one possible way to address these challenges [12].

In an open source business model, much of the value provided to customers will not be provided solely by one company but rather by developers from all around the world who are attracted to working on the open source products. The outside developers may be motivated by the prospect of working with software that solves important problems for them and for others, by the possibility of future gain providing related services and creating related products by the opportunity to increase their own knowledge, or by the ego satisfaction of building an enhanced reputation among their peers [7].

It is important to note that open source business model does not mean non-profit businesses. Open source businesses make millions of dollars every year [4]. There are a number of business models that have been attempted with open source software. The next section will analyze various open source software business models.

7. Support Seller Type Business model

In Support Seller Type business model, the revenue comes from media distribution, branding, training, consulting, custom development, and post-sales support instead of from traditional software licensing fees [4]. This model was first implemented by FSF. It is the most common model among the open source models. Revenues are generated by selling two broad categories of items: physical goods or services. Companies can differentiate themselves by providing more complete and easier to use software distribution, in essence simplifying and improving the user's experience with the open source software in question. Some successful businesses that use open source business models include the Linux kernel vendors such as Red Hat Software. Red Hat sells Linux and various open source software with some limited technical support and consulting through CD-ROM or online presence [2]. Visitors to the Red Hat website will see that many development tools are freely downloadable. The strategy behind the Red Hat idea was that people buy name-brand products even though generic products are widely available at lower or no cost. Furthermore, intellectual property notice at Red Hat website explains that you can also resell the open source product or provide technical support on the same product. Red Hat has gained a reputation for doing business and for helping its customers solve their problems. The company makes most of its revenue from technical support. That is, Red Hat makes the bulk of its profits not from subsidiary products to its free ones, but from the vast service sector that develops around its free product. The products are also available online and customers can download it for free. But if the source code were freely downloadable why would people buy from Red Hat at all if they could get the code themselves for free? The reason is because there is something to be said for the "convenience" of a book and a CD-ROM. Even though people could get the code for free, many choose to buy it from Red Hat because it is simply convenient to do so. Further, developers often complain that, though they are very happy with open source code, their managers will only accept something "in a box" [8]. Although Red Hat's "convenience products" are profitable, they do not necessarily represent the majority of Red Hat's profit [9]. Red Hat makes most of its money in another type of service, one based upon its reputation and experience. Red Hat has gained a reputation for doing good business and for helping its customers solve their problems. Reputation, like status and experience, is vital. Customers often are willing to pay a little extra to make sure that their problems are solved by the best. In the end, hassles can cost more to an organization than a support fee. Who better to hire than the original developers themselves? Red Hat, perhaps more than any other company to date, has done an incredible job at leveraging the product. In fact, they have been so successful that they can afford to maintain a research and development department of six full time developers working on theoretical tangents.
Generation of revenue from the open source product could be done as in the "Support box Sellers" model, i.e., by selling media and services. However, typically the bulk of revenue generated would be through sales of other software products; having the open-source product could increase sales of such traditional products in various ways: by helping build the overall vendor brand and reputation, by making the traditional products more functional and useful, and by increasing the overall base of developers and users familiar with and loyal to the vendor's total product line.

8. Loss Leader Type Business Model

In the "loss leader" model, providing the product makes it more possible that customers will buy other products that are sold using the traditional software business model. In this model, open source software is used to create or maintain a market position for proprietary software that generates a direct revenue stream [8]. Open source client software enables sales of server software or advertising revenue by a portal site. Netscape is using this model where Netscape Communication products are sold while Netscape provides the Netscape Navigator for free. The aim for giving away the source code of a proprietary product is to gain market share. Netscape realized that the trick about software was that once a customer has grown accustomed to a software product he or she is less likely to switch to a competitor's product. In the end, Netscape became synonymous with the World Wide Web and as a result Netscape was able to sell its server product line with great success. By giving away the software, Netscape gained an incredible market share in a short time and was able to sell its server product line with great success. Netscape Navigator essentially ran as a loss leader for the Netscape Server product line. Another example of this business model is Qualcomm who produces Eudora, the popular email program, as freeware in order to sell upgraded versions and Sun's distribution of Java [1].

9. Widget Frosting Type Business Model

The "widget frosting" model is used by companies who are primarily in the business of selling hardware [1]. These companies use open source software to distribute with the hardware at no charge. The enabling software could be driver code, compilers, or complete applications or operating systems. Corel, a hardware manufacturing company, uses Linux for its network systems and reduces the cost of the total system. In this model, most of the revenue comes from selling the hardware, and by using open source software the companies can increase the revenue of the hardware business by providing low cost products. The widget frosting model benefit by having a large development pool and more rapidly changing and flexible responses to customer needs. Apple Computer adopted widget frosting business model by open sourcing "Darwin", the core of the Macintosh operating system server and increased its Macintosh computer sales.

10. Service Enabler Type Business Model

In the "service enabler" model, a company creates and distributes open source software primarily to support access to revenue-generating on-line services [7]. In this model, software is open sourced to create a market position for services. The market building effect of open source can be extremely powerful, especially for companies that are inevitably in a service position to begin with. Digital Creation, a web site design company that specializes in complex database and data transactions, used open source to make its software free. Open source software in this case generates more value as a market builder than as a secret tool. An example would be a company providing a fee-based on-line game service, which created special purpose software to access the service and then distributed the software as open source so that users could not only download the software at no charge but also customize the software.

11. Sell it then Free it Type Business Model

"Sell it then free it" business model is defined as first developing and structuring the software as traditional software and then converting it to open source software when the company reach an appropriate point in the software life cycle where the benefits of developing in an open source outweigh the direct software license revenue it produces [5]. The appropriate licensing model for this type of model would be the BSD style licensing or the MozPL license. This business model helps companies build new revenue streams where the software is in the maturity of its life cycle. However, there are issues of when to give away the source code of the software. If the company converts the software to open source early, it unnecessarily forgoes significant license revenue. If the software product is in the beginning of its life cycle then this model is not appropriate to convert from the traditional model to open source software model.

12. Brand Licensing Type Business Model

In this model, a company makes the software product itself as open source but retains the rights to its product trademarks and related intellectual property and charges other companies for the right to use those trademarks in creating derivative products distributed under the exact same brand [1].

13. Hybrid Business Model

Hybrid business models are the transient models between the traditional and the open source business models [1]. A company using a hybrid business model not only derives revenue and profits directly from selling the shrink-wrapped product sale but also benefits from involving with the open source community. A successful illustration of the hybrid model is implemented by Sendmail. Inc., which released commercial product based on the open source version of the popular Sendmail systems. Sendmail has a hybrid model where commercial product line and open source development process is used together. The commercial line fund additional development resources for open source and for those requirements that are unique to the commercial customer base. Sendmail sells the commercial software that is built on the open source projects.

Another example of the hybrid model is Apple's project on its new operating system, OSX. Apple has a project called Darwin that consists the basis of OSX. Darwin is an open source project and it is free to the community to download. However, OSX is proprietary software where Apple engineers have built their own application over Darwin. Apple is contributing to Darwin because it benefits many ways such as bug fixing, support for peripherals and system code optimization for server to desktop environment. Apple also continues to create new products or improve the quality of the program by open source when selling its proprietary software. As we discussed before, every company can write its own license that specify the terms of the open source project. Therefore, Apple has its own open source license (APLS) that promotes open source development of the software while at the same time allowing Apple to reasonably protect the intellectual property and meet the business goals. Hybrid model is a win-win situation both for Apple and the developer community because developers can make their own modifications and sell it while Apple improves the software quality and uses the skills of the open source community developers.
As seen from the examples, hybrid models help commercial companies to leverage the benefits of the open source software when making money from the traditional business models. It also helps developers to build a commercial product line from the open source software.

14. The Role of Licensing in Business Models

The Role of Licensing in the Traditional Model

Proprietary software vendors have a financial interest in maximizing their return on investment by prohibiting modifications to products, limiting the number of copies a customer can make and restricting their distribution. Licensing in this kind of interest contributes to the commercial product in the form of legal protection. The conventional way to get revenue in exchange for a value of a proprietary software product is to sell the costumers the right to use the software as opposed to selling them actual ownership of the product. Buying software from a store is buying a right to use license rather than possession of the software itself. In the traditional software business model, the companies provide all of the value to the customers and revenue and all profits come in return for that value through the software product.

The legal basis for right-to-use the software is that the specific code and techniques underlying the product are considered to be intellectual property of the developer protected using legal constructs such as copyrights and patents. The owner of that intellectual property, the developer, can control its distribution and use through legally-binding and enforceable contracts, and can charge a fee to individuals and organizations wishing to enter into such contracts.

Considered strictly from a business point of view, right-to-use fee has several advantages, especially for the software vendor but in many cases for the customer as well. First, right-to-use fees can be tailored to work in a wide variety of ways; for example, software can be charged per-user, per-machine, per-CPU (for multiprocessor systems), per-concurrent-user, or for an entire organization or part of an organization (site licensing). Different pricing schemes can also be employed based on the use to which software will be put; for example, using software in support of a particular end use may be done using a separate fee from using that same software in support of a different end use, even though the actual users of the software may be the same in both cases. This allows the vendor to offer preferential pricing schemes for certain customers or end uses as appropriate. For instance, the vendor might allow no-charge unrestricted use by all or some noncommercial customers, or might charge less for software used only internally versus software used to provide services to external users. Second, software fees are independent of amounts charged to the customer for services such as technical support, consulting, and systems integration. For US software companies operating under SEC rules software license fees can normally be recognized as revenue immediately at the time the associated product is shipped; by contrast, service fees can be recognized as revenue only over time as the services are actually delivered to customers. This can be an important consideration, especially for small software startup companies that need to grow revenues quickly in order to satisfy investors and fund growth in operations.

In the traditional software business model, licensing is merely a form of keeping the intellectual property and the copyright. The business plan and the revenue stream drive the software innovation, implementation and the distribution. If the company does not have enough resources to support the development then the software project does not receive enough resources from the company. Only the software projects that have market value would be developed and improved. The licensing has no impact on either the business model or the development process. The traditional business model is market driven. Therefore the licensing plays little effect compared to revenue, profit and marketability.

15. The Role of Licensing in the Open Source Business Model

The value of open source licensing is the guarantee of freedom that the licensing terms provide. No open source product will ever became obsolete because the product's source serves as the foundation upon which future works are derived, and everyone is free to use the source to derive new versions. The availability of free source code makes the topic of open source licensing the primary concern of the open source community. License sets the rules of the game. As discussed in the open source licensing section, every project may have its own open source licenses depending on the special needs of the project. Some projects give the source code and do not require and derivative work to be returned to the community as in BSD License. On the other hand, some license may ask to return all the derived work to the community as in the GPL license. Therefore, licenses play the big role in the open source community. The open source licenses drive the software innovation, implementation and the business models. For instance, open source software with a very strict license would apply to different communities and developers than the

ones with more forgiving terms. The business strategy associated with the licensing is best described as the service industry. This business strategy in which the company sacrifices profits on initial equipment or product sales in order to latch on to long-term service contracts has been applied in countless industries. What a company really wants is to tap into is a steady revenue stream, which will support continued long-term prosperity. Therefore, upfront sacrifice on the equipment sale assures long-term success.

Companies are always looking for ways to differentiate themselves from the competition in order to form a seller's market. However, physical products themselves often have a tendency to be a commodity leading to buyer's markets. In a commodity buyer's market, a company can distinguish itself from its competitors by the quality of service it provides. In fact, many companies simply leave the commodity market to manufacturers seeking the greener pastures of service markets. Higher quality service businesses can charge premium service fees. Open Source is an outstanding candidate to apply the service business model. But Open Source does not just have network effects and a tried and true business model going for it. It has open standards going for it as well. Open Source development efforts invariably employ open standards because of their widespread availability and interoperability, among other reasons. Since Open hackers are not in it for immediate financial reward, there is little incentive to develop closed proprietary technology. Because of the advantages of speed of development and interoperability, it makes much more sense for Open Source hackers to take existing open standards and apply them to the development of Open Source [8]. Additionally, Open Source enjoys the benefits of the religious fervor of people who adhere to open standards strategies. In light of the information presented it can be said that licensing drives the open source business models and it also assures the rapid diffusion and adoption of software in the market. Open Source License also makes the project more attractive because it ensures that the work is protected.

16. The Role of Licensing in the Hybrid Business Model

In the hybrid model, companies build commercial software as an extension to open source projects and company defines the licensing terms [8]. The companies rely on the success of their open source projects when implementing a hybrid model. The hybrid model help the companies to add additional value to its product line by adding new features to existing product, adding new products by using the resources of open source community. However at the same time companies have to make sure that the use of these resources are governed by some kind of licenses. The open source licenses contribute to this process by protecting the rights of the developer and the investment of the company. Companies such as IBM, Sun, and Apple contribute to the open source community and as a return they use and enjoy the rapid adoption and improvement of their product lines. By using licenses companies may have two business models running together. The combined licenses help the companies to keep the original work and improve the quality of the products by participating in open source projects. Companies can design their licenses so that the business model that is already in use does not conflict with the new open source models that area applied. As a result open source licenses enable commercial companies to survive in a fast changing industry environment.

17. Conclusion

In light of the information presented in this paper it is important to note that the traditional model relies on the sale of the product while the license is a part of that business model. Revenue and sales drive the business model, which the licensing is merely a form of legal protection. Licensing in this kind of interest contributes to the commercial product in the form of legal protection. In contrast, the open source software business model licensing drives the business model. It not only protects the copyright and the intellectual property of the software but also aids the rapid diffusion and adoption of the software in the market. The open source licenses drive the software innovation, implementation and the business models. Open source license also makes the project more attractive because it ensures that the work is protected. Hybrid Business model is a way for companies to migrate partially or completely to open source while protecting the copyright and intellectual property of their existing proprietary and forthcoming open source products. In hybrid business model, licenses contribute to this process by protecting the rights of the developer and the investment of the company. It is also important to note that if the software industry moves from the traditional model to the service model, more open source software projects would be sponsored and more open source business models would be in the market in the next few years.

References

[1] Building a Business Model, 2001. Available at http:// www.openresources.com/

[2] Dyson, E., Open Source Revolution, in The Open Source Revolution. 1998, EDventure Holding: New York.

[3] FSF Licenses, Free Software Licenses, 2001. Available at http://www.fsf.org/

[4] Goth, G., The Open Source Market Woos Open Source. IEEE Software Magazine, 2001.

[5] Hecker, F., Setting up the shop: The business of opensource software , 2000. Available at http://www.hecker.org/writings/setting-up-shop.html

[6] O'Reilly, T., Lesson from the Open Source Software Development. The Communication of the ACM, Apr 1999. Vol. 42 No. 4

[7] Raymond, E., The Cathedral and the Bazaar, O'Reilly and Associates, 1 ed. 2001

[8] Sol, S., "WDVL: The Open Source Business Model!" Web Developers Virtual Library. Online. Internet. 4 Dec. 2000. Available at http://wdvl.com/Software/Open/Source/business_model.html

[9] Stallman, R., Free Software Foundation, 2001. Available at http://www.fsf.org/

[10] The Free Software Definition, 2001. Available at http://www.gnu.org

[11] The Open Source Definition, 2001. Available at http://www.opensource.org/

[12] Valloppillil, V. and Cohen, J., Microsoft on the Open Source Software Revolution, 1998. Available at http://www.opensource.org/halloween/

[13] What is Copyleft?, 2001 Available at http:// www.gnu.org