Open Content & The Search for a Better Intellectual Property Model: Lessons from Open Source

Max Preston
mcp8@duke.edu

Rachel Zurer
raz3@duke.edu

December 11, 2003

Abstract

A profound change in the nature of Intellectual Property is occurring in a variety of fields today. These changes mark a shift away from the restrictive, proprietary model that has dominated such fields as music, software, and publishing, to an Open Content model which favors sharing of intellectual property. The Open Content model is most completely illustrated in the arena of Open Source software development, a mature instance of an Open Content domain that has been successful in following a set of principles based on intellectual cooperation rather than competition. Through a comparison with the nature and characteristics of Open Source software development, we predict that the Open Content model of Intellectual Property will be a useful and successful framework for many domains beyond computer code.

1 Introduction

Intellectual Property (IP) issues are one of the most prominent topics of the new millennium. Intellectual Property is defined as "property from original thought protected by law" [12]. Unlike standard physical property, there is no physical result to protect. Conflicts develop over how to protect an intangible object, how to reward creators of intangible objects, and how to grant ownership of these objects.

In this paper we evaluate the viability of "Open Content," (OC) a non-traditional method for dealing with intellectual property issues. This model seeks to alleviate some of the restrictions on creativity inherent in the current/traditional copyright. We explore whether the OC model provides sufficient motivation for contributors to create new works, for creators to publish and distribute those works, and for consumers to obtain them. We also address the question of whether OC can support a sustainable business model. Based on our evaluation of the current state of OC and on our extensive comparison with Open Source Software (OSS), a mature example of the OC concept, we conclude that OC is indeed a viable option for creating and sustaining a community of contributors and users across a variety of purposes and mediums.

To fully understand what is unique about the OC model, we must first examine the basis for and current state of copyright law in the United States. Only then can we understand how OC works within the traditional framework to allow creators to explicitly relinquish some of the rights they would otherwise retain. To evaluate the impact that this “some rights reserved” concept has on intellectual property, we will look at the Open Source (OS) model of software development, which is based on many of the same concepts as OC and can be considered a sub-category of the OC model. By examining what makes Open Source development viable and extending those conclusions to OC as a whole, we will show that the OC model does indeed have the potential to support a vibrant and productive community of contributors and users.

2 The Basics of Copyrights

> [The Congress shall have power] “To promote the progress of science and useful arts, by securing for limited times to authors and inventors the exclusive right to their respective writings and discoveries;” U.S. Const. art. I, Sec. 8, cl. 8.

The above passage of the Constitution gives Congress the authority to create laws with the goal of promoting progress in science and the arts. These laws confer a temporary monopoly over a work to its author or inventor, allowing her to profit from her creativity. The possibility of profit in turn encourages authors and inventors to continue producing new works. The idea behind the laws is to give producers some incentive to share their works with the public, rather than to reward them for their efforts or labor [38].

Copyrights apply only to tangible *expressions* of ideas, and not to the ideas themselves. Additionally, copyrights apply only to original works. If a work is original enough to qualify for a copyright, it is protected from the moment it is fixed in a tangible form. Registration is not necessary for a work to be protected [13]. Currently, copyrights last seventy years after the death of the author, or ninety-five years for “work made for hire”. Copyright holders are granted several exclusive rights, including the rights to reproduction, distribution, and to “derivative works” [40]. However, the public reserves several limitations on these exclusive rights, known as fair use. These fair use rights include “reproduction for purposes such as criticism, comment, news reporting, teaching,...scholarship, or research” [40] (Title 17, Clause 107).

2.1 Problems with the Current State of Copyright

One of the modern intellectual property controversies is over the intended beneficiary of copyright laws. Copyright has been described as a balance between the rights given up by the public and the allowances granted to the creators [38]. What exactly characterizes a balanced system has become a point of contention between those seeking to profit from copyrighted works and the public who seeks to maximize their benefit from those works. Each side of the controversy tends to portray themselves as the intended beneficiary of copyright law. However, the original intent of copyright law, as we have seen, was to maximize public good. Therefore, it is the public and not the copyright holder whose interest should be primary in decisions regarding intellectual property rights [38].

Nevertheless, current changes in copyright law have tended to afford more protection to copyright holders. In 1998, Congress both extended the terms of copyright and enacted the Digital Millennium Copyright Act (DMCA), which places severe restrictions on the use of digital copyrighted material, to the point disallowing activity that is traditionally permissible as fair use. These developments have raised concerns with many individuals who feel that the public interest is not the driving force behind these laws. As Lydia Pallas Loren, a law professor at Lewis and Clark College asks, "Will the significant cost to the public of [an] increased monopoly be outweighed by the additional works that will be created and disseminated as a result...?" [19]. There are many who feel that this cost will *not* be outweighed by any benefit purported to come from stronger copyright laws. It is among this group of dissenters that some have started to look for a different method for dealing with intellectual property.

2.2 The Public Domain

"The public domain is the figure and copyright the ground." –James Boyle [5]

An important concept when discussing intellectual property is the area of the public domain. The term itself originally referred to public lands, accessible to all. In the recent times, however, public domain has been extended not just to physical property, but also to intellectual property. Public domain, as defined by Yochai Benkler is "the range of uses of information that any person is privileged to make" [2]. In other words, the public domain is the body of knowledge and intellectual property that is both free from exclusive rights and free in terms of cost. The importance of the public domain lies in the idea that from this common pool of knowledge, mankind is able to draw a useful basis from which to progress. It is from this pool that Disney drew the stories for many of their cartoons. It is because the Polio vaccine was left in the Public Domain and not patented that the disease was able to be treated so quickly. The public domain easily allows anyone to improve upon what has already been done. As Isaac Newton said in 1676, “If I have seen further, it is by standing on the shoulder of giants.”

Public domain, together with copyright, forms a dichotomous system that covers all of intellectual property. Copyright covers the protected uses of intellectual property; public domain covers the unprotected uses. On the boundary between them lie licenses. Within the range permitted by copyright law, the type of license on a piece of intellectual property can either expand or restrict the public domain or expand or restrict the copyright. As such, licenses can play an integral role in the accessibility of knowledge.

2.3 Open Content: A Copyright Alternative

Not everyone agrees with the specific balance between creator’s rights and the public good embodied in current copyright law. Concerned that the increasingly restrictive copyright protections provided by default when a work is created are harmful to the public good, a movement has emerged to provide a mechanism for rights holders to release some of those rights. Dubbed “Open Content,” the goals of this movement are to “use private rights to create public goods: creative works set free for certain uses” [6].

Through its licenses, which rights-holders can choose to apply to their works, Open Content seeks to create a better balance between public domain and copyright. The balance which Open Content seeks to create has been referred to as a “commons.” This commons, “protected by a liability rule rather than a property rule” [17] allows users to benefit from ideas and innovations without having to ask for permission or tiptoe through a legal net. It protects innovation and freedom while only adding the bare minimum of property protections. It encompasses many arts: music left open for sampling, video left open for editing, educational resources free for all to study and add to, source code for a community to refine and contribute to. In the case of computer code, as James Boyle put it, open content “attempted to build a living ecology of open code, where the price for admission was your commitment to make your own incremental innovation part of the ecology” [5]. The Open Content movement does not intend to make everything a part of the public domain, but rather to bring more freedom than standard copyright and to encourage innovation through increased privileges of use.

3 Introduction to Open Source

The open source (OS) method of software development represents the most mature and extensive example of the Open Content model to date. Open Source software (OSS) is software in which both the source code and the binary program are freely available to the public, for examination, use, and modification. Because of this freedom to modify and build on others' work, OSS allows programming to occur in a distributed fashion. Some advocate open source because they feel it produces a better product due to this bottom-up collaboration, others because they view it as an issue of freedom. In any case, software developed on the Open Source model provides a real alternative to proprietary software, in which manufacturers retain their full rights to the product. We will examine the particulars of the Open Source model in order to gain a better understanding of the challenges and possibilities facing the burgeoning Open Content movement.

3.1 Defining Open Source: Software Licenses

To really understand Open Source software and how it relates to Open Content, we must examine what qualifies as OS and the way that various OS licenses use and modify the framework of copyright law on which they're based. In contrast to Open Source, proprietary software reserves all rights for the owner, making it illegal for users to distribute the software or create derivative works. Some end-user license agreements included in proprietary software go even further in restricting the rights of the user. By contrast, OSS is explicitly based around the privilege to modify and distribute code, with various conditions. Most of the popular or mainstream software today is proprietary (everything made by Microsoft, for example), though as we shall see OSS has had its share of success.

The term "Open Source" is in fact a certification mark owned by the Open Source Initiative, which provides a definition of what exactly qualifies as Open Source software. The main requirements for Open Source are:

- The software must be redistributable for free.
- The source code must be freely available .
- The user must be able to make modifications to the software and to freely distribute those modifications.

- All users must have the same privileges, i.e. there must not be any discrimination against persons, groups, or fields of endeavor.
- The above privileges must be technology-neutral and not specific to a product.
- No other software should be restricted by the license. [26]

These conditions are all made explicit in the licenses under which the software is distributed. The Open Source Initiative lists over 40 OS-approved licenses, but a few of them stand out. The most liberal of the licenses is the Gnu General Public License (GPL), characterized as "copyleft", because "it's sort of like taking copyright and flipping it over" [38]. Not only does the GPL explicitly allow modification and distribution of the code, but it also requires that all derivative works be similarly licensed, thus ensuring the perpetual freedom of the software. Other variations of Open Source licenses include the LGPL (used mainly to allow libraries to be linked into non-free programs), BSD (developed by programmers at University of California at Berkeley), MIT and Mozilla Public License (created by Netscape to distribute an open-source version of their browser). These and the other Open Source licenses have some differences in their exact wording and what they allow, but they are all based on the same general principles. In every case, software licensed under the OS model allows for free modification and distribution of the software, which places full power in the hands of the user to change the software to fit his or her needs and tastes.

4 Examining Open Source

In order to make forecasts and predictions as to the viability of the Open Content model for intellectual property, we will examine the various successes, failures, strengths, and weakness of Open Source software. We will base our exploration on the framework for analyzing Open Source outlined by Joseph Feller and Brian Fitzgerald in their cogent analysis of the Open Source paradigm [8].

4.1 What is Open Source?

We have already seen what defines a project as Open Source: its license must adhere to the conditions set forth in Section 3.1, allowing free modification and distribution of the source code. SourceForge.net, a major clearinghouse for hosting OS projects, features tens of thousands of OS projects at any given time, and provides a sample of the type of development currently occurring in the Open Source movement. The top categories (measured in order of the number of projects) include: Internet-related programs (servers, browsers, content, site management, search); System programs (networking, operating systems, hardware, system administration); Software Development (build tools, code generators, compilers); Communications (e-mail, chat, file-sharing); and Games (role-playing, simulation, strategy, etc) [35]. Only these last two categories have much potential to appeal to the general computing public. The other top categories reflect the fact that Open Source projects have historically tended to be "operating and networking systems software, utilities, development tools, and infrastructural components" [8]. Other types of Open Source projects include desktop productivity (e.g. word processing software) and entertainment programs (e.g. media players).

4.2 How does Open Source happen?

The OSS development model is a distributed process that makes use of the real-time, geography-neutral capabilities of the Internet. Eric Raymond, a leader in the Open Source movement, goes so far as to declare that "cheap Internet was a *necessary* condition for the [current] model [of Open Source development] to evolve" [emphasis added] [29]. He points out that the birth of the Linux operating system (one of the best-known and most successful OSS projects), and the growth of OS development in general, coincided with the "explosion of mainstream interest in the Internet" [29]. The Internet provides an affordable, practical, and efficient forum for people to work together across widely dispersed geographic regions. The success of the Open Source model rests on this foundation of a practical communications medium.

Given that a workable communications medium (like the Internet) exists, the software development process is usually centered around a small group of core developers who coordinate the contributions of a larger group of distributed programmers. The most important job of the core group of a project (which can be a team or a single individual) is coordination and diplomacy. Such a leader's role is to "capture the benefits of having lots of co-developers without having a project collapse into a chaotic mess" [29]. This task involves keeping contributors working toward a cohesive goal, avoiding splintering of a project into factions. Linus Torvalds, initiator of Linux and perhaps the best-known of such leaders, fills the role as "an ambassador, roaming his virtual world and exerting his influence to prevent technical fights from devolving into sectarian battles" [33]. Contributors to Open Source projects usually submit their work to this core group of "owners" or "maintainers" who then incorporate it into the latest release of the software, making it available for other contributors to review and modify. The process represents a continual cycle of release and modification throughout the life of the project.

4.3 Who is Involved in Open Source?

Most Open Source developers are highly-motivated professional coders who work on OS projects in their spare time [36]. Eric Raymond asserts that due to the competitive nature of OS, "its culture only accepts the most talented 5% or so of the programming population" [29]. Whether or not this is true, it does appear that a small group of programmers (the top 10% in terms of quantity of contributions) is responsible for the majority of work on OS projects (72%), while the majority of contributors (75%) only make one contribution [16]. The voluntary nature of Open Source and the harsh peer review that code undergoes ensures that those programmers who are involved are motivated and talented enough to produce work of a relatively high quality. Recently, large corporations have become involved in hiring programmers to work on OSS[1].

At its inception, most users of OSS have were developers as well. As OSS has matured, however, and businesses such as RedHat have focused on providing OSS suitable for use in corporations, the use of OSS has expanded into many mainstream enterprises. The Open Source web-server Apache currently has more than 60% of the market share, according to Netcraft.com. RedHat advertises that corporations such as Amazon.com, British Petroleum, and DreamWorks use their versions of the Open Source Linux operating system. The United States Department of Defense is a major user of Open Source products [36]. Finally, there is a significant trend toward Open Source products in many developing countries, with China at the forefront of this movement. Using OSS allows these countries political freedom from any influence the United States might have over proprietary software: "'There have been rumors for years that US intelligence agencies have persuaded all North American suppliers to put 'back doors' into software' to allow government cyberspies to keep tabs on sensitive information abroad...'Vendors claim it's not true, but those reassurances do not stop the professional pessimists' who head foreign intelligence services" [36]. By using OSS, governments not only get the benefits of a world-wide resource pool of programming expertise at an extremely low cost relative to proprietary products, but the assurance that they are in complete control of their software. With the backing of such major powers as China, India, and Brazil [28], Open Source development is poised to receive a push toward simpler program interfaces

[1]The Open Source Development Lab is a notable example. It is a corporate-funded consortium created to help propel work on the Linux operating system. RedHat, a Linux distributor, is another company which bases its business around Open Source and employs programmers to work on OSS. It has proved extremely successful in its service-focused business model which emphasizes reliable and flexible software.

and more broadly appealing software that could launch OS products even further into the mainstream.

4.4 Why Contribute to Open Source?

There are three main categories of reasons that motivate programmers to adopt an Open Source model or to contribute to OS projects: political; socio-economic; and technological. Any given contributor will likely be working from some combination of these three motivations, which can shift in relative importance as conditions change. Some experts in the field, namely James Boyle and Yochai Benkler, argue that the question of "why" programmers engage in Open Source does not matter. They contend that given an infrastructure that supports such development,"a random distribution of incentive structures in different people" is enough to ensure that "they *will* do it." [5]. Despite their claim that "if you build it, they will come," it is informative to examine the specific reasons contributors involve themselves with Open Source so that we may later explore how such motivations apply to Open Content on a broader scale.

4.4.1 Political Motivations

The political ideals expressed through Open Source programming are in fact most explicitly captured by a related but separate phenomenon, the Free Software Movement. Free Software (FS), though in practice functionally similar to OSS, differs from Open Source in that it regards the privileges guaranteed through the OS licenses (see Section 3.1) as fundamental and inalienable rights. The GPL in particular is an expression of these beliefs, since it dictates that all derivative software must also be free. Richard Stallman, leader and founder of the Free Software Foundation and the GNU Project (two important FS institutions), is adamant about the differences between OS and FS, and would likely object to the current conflagration of the two [37]. Nevertheless, since Free Software (in terms of essentially everything but philosophy) can be seen as a subset of the Open Source development model, it is useful here to examine the tenets of the Free Software Movement as they relate to political developer motivations.

> "So imagine what it would be like if recipes were packaged inside black boxes. You couldn't see what ingredients they're using, let alone change them, and imagine if you made a copy for a friend, they would call you a pirate and try to put you in prison for years. That world would create tremendous outrage from all the people who are used to sharing recipes. But that is exactly what the world of proprietary software is like. A world in which common

decency toward other people is prohibited or prevented" [37]

This metaphor, imagined by Stallman, embodies the ideals that motivate programmers to adopt Open Source for political reasons. The prohibition on sharing inherent in proprietary software is distasteful to many and runs counter to some of the values held by the prevailing Judeo-Christian ethic. Stallman summarizes this belief by saying, "if you want a better society, you've got to work to encourage the spirit of sharing" [37]. It is thus for the advancement of social good and a community ethic that some programmers adopt Open Source.

Even for those who are less concerned with the building of community and the values of sharing, the freedoms given by OSS provide a strong motivation to engage in its development. A popular slogan that helps explain the "free" in "Free Software" reads "free speech, not free beer". Hidden within this catchphrase is an appeal to the deeply-rooted democratic ideals of personal freedom that are important to many individuals, particularly in a Western, individualist, type of society. The restrictions proprietary software puts on one's ability to use the code as one chooses or to personalize it to meet individual tastes and needs runs directly counter to these libertarian ideals of individualism and freedom. The personal freedom guaranteed by OS products provides another political motivation for contributors to adopt the Open Source model.

4.4.2 Socio-Economic Motivations

The economics behind engaging in Open Source development can be analyzed on two different levels. As more companies whose business models are founded around the Open Source paradigm emerge, it becomes realistic to examine what the pay-off is for these companies to break from the traditional methods of software sales. Until recently, however, the idea of making money off of OSS seemed far-fetched and fanciful; the adoption of an OS-focused business model is an important but relatively late development. For years before programmers had any hope of receiving any of the traditional incentives of salary and benefits to work on OSS, Open Source had contributors. Thus the more fundamental question, as asked by economists John Lerner and Jean Tirole, is "Why should thousands of top-notch programmers contribute freely to the provision of a public good?" [16]. We will focus our examination on those socio-economic motivations that spur contributors even in an environment where contributors don't expect to get paid directly for their work on developing OSS.

One of the fundamental characteristics of the OS development community is the set of explicit and implicit rules surrounding attribution and recognition of code. While by the very nature of Open Source, no one holds any exclusive rights over the products they produce, there is a very strong pressure toward ensuring those responsible for a product receive proper recognition. This force is made explicit in some of the licenses used in OSS (until recently, for example, the BSD required that all derivatives give credit to the Berkeley regents no matter how distant the relation). The pressure toward recognition is also important in guiding some of the more implicit rules of the OS community, for example those regarding transfer of responsibility for a project [30]. Because of the norms protecting recognition of contribution, and because of the strict meritocratic manner in which code is evaluated, the OS model functions on a currency of reputation.

Whether consciously or not, a major motivation behind contributing to an Open Source project is to further one's reputation among one's peers [30]. Occasionally, a programmer's prestige in the OS community can be a "stepping stone" to positions in other arenas, as for Linus Torvalds (father of Linux) [33] or Larry Wall (originator of the Perl programming language) [16]. More often, however, it is within the OS community that a programmer seeks recognition, as "good reputation among one's peers is a primary reward" and "prestige is a good way (and in a pure gift economy, the *only* way) to attract attention and cooperation from others." In the absence of any material goods to exchange and without any pre-existing hierarchies or loyalties, "peer repute" becomes the only way in which to gain status and thus a prime motivating force for contributing to OSS [30].

4.4.3 Technological Motivations

While from a user's perspective OSS often has many technological advantages (discussed in Section 4.5.3), from an contributor's standpoint technological motivations related to Open Source take on a much more personal aspect. Programmers inherently have a personal relationship with code they've participated in creating, and it is the Open Source model's freedom to maximize this relationship that encourages contributors to adopt it. An important feature of OS projects is that contributors get to choose exactly what problems they wish to work on. This freedom of self-direction both allows for a greater sense of ownership over the final product and a greater ability to focus on areas of interest than in proprietary programming. Eric Raymond points out that the personal relationship that programmers feel with their code is true by definition: "People for whom ['the pure artistic satisfaction of designing beautiful software and making it work'] is not a significant motivation never become hackers in the first place, just as people who don't love music never become composers" [30]. Given that the self-selecting group of OS contributors tends to take pleasure in their code craftsmanship, and that they are free to work according to their own tastes, it is no wonder that developers often cite “the joy of hacking” as their main motivation for contributing (the though effects of reputation discussed in Section 4.4.2 should not be ignored). Works of software often start “by scratching a developer’s personal itch” [29], such that programmers gain an immediate reward from the successful implementation of the project (beyond the joy of a job well done). Once programmers have done the work to “scratch an itch,” their decision to make their work available as Open Source can be motivated by the political and socio-economic reasons discussed above, or by technological considerations. By making their contributions available for others to both use and enhance, they trade whatever benefits they might have received for keeping it closed for the help and work that others can contribute to improving and maintaining the product (and thus increasing its quality and usefulness to the author) [31].

4.5 Why Use Open Source Products?

Much as there are many categories of motivations for developers to contribute to OS projects, there are several different types of reasons that users might choose OSS products over proprietary software. We will briefly examine these motivations, which can be fit into the three general categories of freedom, cost, and quality.

4.5.1 Freedom

The same political and philosophical tenets that can motivate developers to contribute to OS projects (see Section 4.4.1) can also motivate users to choose OSS. Choosing Open Source products (and by consequence *not* purchasing proprietary products) can be an ethical and political statement against the restrictions inherent in proprietary software, a sort of boycott. Yet for many users who choose OSS, the appeal is not so much freedom in a political sense as in a practical sense. Because of their access to the source code, users of OSS have complete control over the future of their software. If they want a certain feature, they can write it, or hire someone to do it for them. They are freed from any dependence on a third party. Not only does this freedom allow for total customization, it also provides a sort of insurance for the future (“future-proofing”) [31]. Users of OSS are guaranteed (through their access to the source code) that no matter what happens to the original provider of the software, they will have the option and ability to keep the software up to their current needs (by porting it to new systems, for example).

User of OSS need never fear obsolescence, since it is within their power to prevent it.

4.5.2 Cost

An obvious incentive for using OSS is that, by definition, it is always available for free. Users, especially businesses, will often pay for a convenient or certified package of OSS, but even these services are often cheaper than proprietary software. For home users, small businesses, and developing nations, the low-cost availability of OSS can be a significant motivation for using it [28].

4.5.3 Quality

Advocates of the Open Source development model claim that not only does Open Source development occur quickly and cheaply, but it produces results of a superior quality than the proprietary methods of software development. Eric Raymond explains this phenomenon as "Linus's Law": "Given enough eyeballs, all bugs are shallow" [29]. His claim is that the Open Source model of having unlimited users who are also free to be developers creates an environment where problems with the software are found and fixed thoroughly and quickly. The Open Source system is equivalent to a system of massive peer review. Studies have for the most part found that OS code is as good or better than equivalent proprietary programs.[2] Additionally, problems found in OSS tend to get fixed much more quickly than problems in proprietary software, especially if they are deemed critical or high-priority [1]. While the findings on the quality of OSS relative to proprietary software are not conclusive, the belief that OSS is better can motivate users to adopt OSS solutions.

5 Open Source Case Studies

To enhance and clarify our exploration of the Open Source model, we will look at several major Open Source projects to see how they fit within the framework we have just established. We will consider the nature of the project, how the project came about, who was involved in its development, and who uses the software and why. We will evaluate the success of the Open Source model in each case. Some

[2]One study found that "The open-source implementation of TCP/IP in the Linux kernel clearly exhibits a higher code quality than commercial implementations in general-purpose operating systems" [34]. A later study by the same group "found that the latest version of Apache Web server is comparable in quality to its commercial brethren" [4]. A security analysis of several operating systems found the Open Source OpenBSD to be far more secure than both the Open Source Debian or the closed source Solaris systems [25].

of the criteria we will examine include security, reliability (including number of bugs), and adoption (whether the software is widely used and why/why not). By looking more concretely at some major OS projects, we will have a better base for determining what characteristics are involved in making a project successful, and whether those traits can carry over to other forms of Open Content projects.

5.1 Linux

The Linux Operating system, an open-source version of Unix, is touted as the prime example of the distributed model of development that characterizes Open Source. It is widely used and represents some of the commercial business models possible using OSS.

5.1.1 Brief History

Linux began in 1991 as a project of Linus Torvalds, who was a college student in Finland at the time. He wanted to create an operating system that he could run on his home computer that would be as stable as Unix, a widely used proprietary operating system. He based his project around an existing but unsatisfactory OS operating system called Minix. After completing a rough version of the system, he made it available online and posted a message to a Minix users group announcing that the software was available for use, modification, and feedback. Several months later, Linus added a compression feature to Linux in response to a request from a user, which put Linux ahead of similar operating systems and encouraged its adoption by others. This led to a cycle of increasing returns, as more people started using and contributing to the operating system. As more developers and users became involved in the project, Linux became more useful and more functional, thus attracting even more developers and users. Within a couple years, Linux was stable and popular enough for entrepreneurs to form a company whose business model was based on it (see discussions of RedHat in Section 4.3). Version 1.0 was released in 1994[3]. Today, the operating systems is used by more than 18 million people worldwide and provides the focus for the drive toward adoption of OSS [33].

[3] In OS programming, a version number of 1.0 or higher usually indicates that the owners of the project stand behind the quality of the release, and is a significant milestone in the life of a project.

5.1.2 The Developers

Linux was the first major Open Source project to really take full advantage of the capabilities the Internet offered. All of the early development was done over the Internet, with contributors widely dispersed geographically. Despite this geographic spread, the Internet allowed development to continue at a fast pace. "Early and frequent releases are a critical part of the Linux development model": contributions were submitted and new versions released some times as often as several times a day. At least at the beginning of the Linux development, there was no clear differences between users and developers. The combination of many contributors and frequent contributions kept the project moving forward quickly and kept contributors interested in and challenged by the project [29]. Today, contributors include paid programmers working for major corporations (IBM is a notable example) and there is even a non-profit organization (the Open Source Development Lab) "dedicated to accelerating the growth and adoption of Linux" [15].

5.1.3 The Users

Reliable data on usage and adoption is difficult to find for an operating system that can be freely copied or downloaded. However, RedHat alone has registered over a million systems, and a 2002 study predicted that "Linux will run on 45% of new Intel based servers by 2006 or 2007" [32]. Linux is touted as being both more stable and more secure than its proprietary competitors (including Windows and Unix), though current quantitative data evaluating these claims is again hard to come by. Though Linux has a strong footing in enterprise and server sectors, it is less widespread on home-computers as desktop software. One of the major complaints preventing Linux from becoming widespread with desktop users is that it is not as user-friendly as other systems (Windows in particular) and it doesn't support as wide an array of applications. However, whether or not Linux succeeds in taking over a significant portion of the desktop market, or even in growing its market share in other domains, when evaluated from the point of view of functionality and adoption in general, it has been an extremely successful demonstration of the power of the Open Source method.

5.2 Apache

Apache is a Open Source web server that is generally touted as another key success of the Open Source movement. According to monthly surveys on Netcraft.com, Apache is used to host more than 65% of respondents' web sites. It's share of the market has steadily grown since its launch in 1996 [1]. In terms of adoption, Apache is an example of a wildly successful Open Source project, as it has come to dominate its niche in the market. We will examine the process that led to its development and the factors involved in its success.

5.2.1 Brief History

The Apache project began in 1995 when a group of eight programmers decided to pool their resources and work collectively on improving the currently available web servers. The first version of Apache was based on an HTTP daemon developed by the National Center for Supercomputing Applications (NCSA) which was no longer being maintained centrally and had thus spawned many disjointed improvements. Using a small, private e-mail list, the developers worked to collect and organize all the useful patches and extensions they could find and released the first version of Apache in April 1995. During the next year, they worked to reformulate the base code and add functionality, features, and documentation. In December 1995, version 1.0 was released. Within a year, Apache was the leading web server based on Netcraft surveys [9].

5.2.2 The Developers

Those involved with the Apache project formed an informal group call the Apache Group (AG) to guide the development of the project. Those involved in the AG were all professional developers with other jobs, who volunteered their skills to work on the project in their spare time. The original members were webmasters who stood to gain personally from a better web server (i.e., they were all users as well as developers). As the project grew, almost 400 developers contributed code.[4] Another 3,000 people submitted problem reports (i.e. feedback on the program). Despite the large numbers of people involved, only about 15 people contributed about 85% of the code. The contributions of this core group were disproportionately additions of new functionality, whereas the other hundreds of developers contributed mainly fixes or improvements to existing code [1]. Apache is currently supported by the Apache Software Foundation, a non-profit corporation whose members (selected by meritocracy [i.e. amount and quality of contribution]) vote on issues surrounding the software [9].

[4]Figures were accurate as of May, 1999. The number has certainly grown since then, but the older figures remain informative as to the scope involved in bringing the project to its mainstream status.

5.2.3 The Users

"We want to see Apache used very widely – by large companies, small companies, research institutions, schools, individuals, in the intranet environment, everywhere." -The Apache Software Foundation [9]

With such a large portion of the market share under their belt, it seems that the dreams of the Apache developers have come true. Such major players as the White House, Amazon.com, Apple, and Harvard and MIT Universities were hosting their websites with Apache at the time of writing. Apache has the advantage of being available for free, which means that individuals or small businesses with limited resources have an incentive to use it as opposed to a higher-priced competitor. Additionally, due to the high number of developers and users involved in the project, Apache has a very quick response rate for resolving problems, especially those with high priority; half of all reported problems are solved within a day [1]. The software has been shown to be at least comparable in quality to its proprietary competitors [4]. Through a combination of these factors, Apache has proved to be an obvious choice for many individuals and businesses large and small.

5.3 Lessons from Open Source

There are many theories as to why the distributed, collaborative development permitted by Open Source software works. Conventional logic says that more programmers does not equal better or faster programming. How can it be, then, that open source software, with potentially hundreds of developers, produces satisfactory products? Eric Raymond proposes one theory: "The [Open Source] world behaves in many respects like a free market or an ecology, a collection of selfish agents attempting to maximize utility which in the process produces a self-correcting spontaneous order more elaborate and efficient than any amount of central planning could have achieved." [29]. Open source software development is thus a "complex emergent system" in the sense that a sophisticated product emerges through allowing elements in the system (programmers) to be guided by purely local motivations (as discussed in Section 4.4) rather than imposing global constraints. Some of the key characteristics of the system that emerges and of the conditions in which it emerged include: a practical medium of communication (i.e. the Internet), a core group (with flexible membership) responsible for coordinating the progress of the endeavor, and a project kernel or hook interesting enough to get people involved with and motivated towards the project. These features may be specific to software projects, but as lessons they are helpful to keep in mind as we examine the broader context of Open Content in general.

6 Expanding to Open Content

As of yet, we have discussed much of what makes Open Source both a viable and productive form of code production. We have analyzed the motivations behind the Open Source movement and what makes this movement work. We have discussed how to classify a project as Open Source, and looked at some examples of Open Source. From the examination of Open Source a question naturally develops: Can Open Content be made to fit the same structure as Open Source? If not, what structure would define Open Content as a whole, and how would that relate to the Open Source movement? The first part of this paper focused on Open Source as the most developed form of Open Content today. In the remainder of this paper we will develop a model to govern all of Open Content, and suggest a path to viability. We will discuss both the motivations for Open Content, as well as methods of implementation for Open Content.

6.1 Open Content - The Commons

We had previously discussed the requirements for something to be considered Open Source Software. It is important, when expanding to Open Content, that a similar list be created so as to decide what we are discussing. Therefore, we offer the following requirements for something to be considered Open Content:

- The content must be redistributable for free.
- The user must be free to make modifications or corrections to the content at no charge.
- All users must have the same privileges

It is important to realize that although Open Content takes place in the commons, not everything that takes place in the commons is Open Content. In Open Content, the user has a certain amount of freedoms to interact with the content that is not always present in everything that takes place in the commons. Yochai Benkler refers to a broader term called "commons-based peer production" [3] to describe anything that is created through the commons. This term encompassed Open Content such as music or educational resources, but it also covers programs such as SETI@home of NASA Clickworks. In the latter programs, the user does not have true access to the content, and isn't really free to make modifications. Essentially, not all the users have the same privileges, as the sponsors (SETI, or NASA) reserve the ultimate privileges of modification, checking, analysis, and they are the only users who see the combined results of everyone's work. It is important to acknowledge this second group's existence, as it adds to the commons, but it does not qualify as Open Content and thus we will not focus on it while discussing Open Content's viability.

6.2 How does Open Content Happen?

There are two structures that Open Content can take: one of which is exemplified by Open Source and the second is exemplified by Open Music. The first structure is highly centralized, as already explained, where a community of people is working toward producing a definitive creation led by a group of "owners" or "moderators." In Open Source, this community is working toward creating a specific program such as a Word Processor or an Instant Messager Client. This centralization can be seen in other forms of Open Content as well, for example the Open Directory Project [23] or Wikipedia [41], where a community continuously adds to and edits an Internet directory and an encyclopedia, respectively. Much of Open Content, however, lacks this centralization. In much of Open Content, the common goal of the community is to create a forum in which content can be freely accessed and used. In this situation, there are usually no "moderators," and beyond creating this forum, there is no greater goal of creating a specific project. Open Music, as exemplified by Magnatune [20], is a concept in which music is written and distributed in a manner similar to Open Source. However, there is no specific goal after that contribution. Communities do not work together to write a Techno song, but they do make their music "open" should someone wish to sample their music to create a Techno song. This second, less structured format of Open Content is usually associated with the arts. The purpose and meaning behind art is not usually something that can be agreed upon by a community as a whole. Therefore, the works of art are left open for anyone to interpret, edit, or use in the manner most meaningful to them.

6.3 Who is Involved in Open Content?

Similar to Open Source, the people involved in the Open Content movement are currently a small percentage of the larger overall community. However, also similar to Open Source, many of the developers of Open Content tend to be relative experts in their fields. Open Content is in no way a new concept. Academic researchers have for years practiced a type of Open Content, where their work is shared and becomes common knowledge in the community. "It is easy...to forget that information production is one area where we have always had a mixed system of commercial/proprietary and non-proprietary peer production" [3]. We could point to Isaac Newton's quote already referenced about building on the discovery of others (see Section 2.2). Or we could point to the famous quote by the creator of the polio vaccine, Jonas Salk: "There is no patent. Could you patent the sun?" Either way, peer-production or Open Content has long been a common practice among higher academia. In addition, Open Content in some form has always been practiced among the non-academic community as well. Think about how often people discuss the world around them or discuss what they are working on. People who are passionate about a topic tend to find other people who are passionate about the same thing with whom they can discuss. It is from this passion of the everyday community that Open Content draws its strength. Open Content does not just draw from the experts of academia, but also the "experts" of the community who take it upon themselves to become knowledgable in a topic they enjoy. These community experts are certainly a fringe group, but the group as a whole creates a large pool of creativity from which to draw.

The users of Open Content tend also to follow the pattern of Open Source. Many of the users of Open Content tend to be former or current developers of Open Content. Proprietary Content has much more money to spend on advertising than Open Content currently does, so it is not often that a user will stumble onto Open Content by accident. Usually a person will have to actively seek out Open Content in order to find it. Once they find one Open Content project, however, they are often referred to countless other projects. In the pattern of increasing returns also important to Open Source software projects, as more users get involved Open Content is slowly becoming more common knowledge over time and its user-base is by consequence growing. We conjecture that following the pattern of Linux's development, for example (see Section 5.1.1), the involvement in Open Content will grow exponentially as more and more users get involved and recruit still other users and contributors.

6.4 Why Contribute to Open Content?

The motivations of Open Content developers are essentially the same as those for Open Source. Open Content developers also have motivations divided into three categories: political motivations, socio-economic motivations, and practical motivations. We won't cover these in too much depth, as most of their content has been covered already (see Section 4.4), but we will generalize the arguments to make them applicable to all of Open Content.

6.4.1 Political Motivations

"Founded by Musicians, for musicians. No major label connections. We are not evil."

–Magnatune.com [20]

Just like in Open Source, many contributers to Open Content have a certain image in their mind of democracy and the freedoms it entails. To these contributors, "the free flow of facts and information is...vital to a successful democracy" [27]. Open Content creates this flow of information and further guarantees these freedoms through licenses. For them, Open Content is a return to the view held by the founders of American democracy [27], and thus holds the advantage of both political and moral superiority.

6.4.2 Practical and Socio-Economic Motivations

When discussing Open Source from the socio-economic perspective, we focused mainly on the self-serving aspects of taking part in the Open Source movement. We discussed the role that building prestige or "peer repute" plays. Here we wish to abract that role slightly and perhaps delve more into the view mentioned earlier as presented by James Boyle and Yochai Benkler.

Certainly hedonistic gains of repute plays a large role in the development of Open Content, but certainly there are many other factors as well. The pleasure of creation also plays a role in the development of Open Content as does seeking the benefits of outside review. In addition, though, just a passion for a certain topic is important. Whatever the motivation of a developer, it must be strong enough to overcome the cost of working on an Open Content project (as in any work). The advantage of Open Content and peer-production in general is the division of work among many people. By dividing work among many people, the costs for each person is drastically lowered, meaning less cost to be covered by motivations. This trend leads Benkler to develop a corollary to "Linus's Law:"

> "Given a sufficiently large number of contributions, direct...incentives necessary to bring about contributions are trivial" [3].

Given the medium of the Internet, where a nearly limitless number of people are capable of contributing to a project at once, the labor can be divided down to relatively minute amounts. Which returns to Boyle's argument: Given enough contributors (via the Internet), people will find the socio-political motivations they need to overcome their relative minute costs. [5]

6.5 Why Adopt Open Content?

The users of Open Content also have a number of motivations, similar to Open Source. These motivations can be divided into three categories: freedom, cost, and quality.

6.5.1 Freedom and Cost

The freedoms granted by and cheapness of Open Content are very similar to Open Source. The users have the freedom to modify or redistribute the content. That means the freedom to edit texts, remix music, make collages out of photographs, or to simply make a mix cd with the music downloaded from the Internet. Also, by the nature of Open Content, it is available at no cost. Both of these are similar to Open Source and provide strong motivation in and of themselves, particularly in the context of the high-priced, highly-restricted closed content alternatives.

6.5.2 Quality

The benefit of quality for Open Content is not immediately visible in comparison to Open Source, so we are addressing this separately. In Open Source, the massive number of contributers allows for massive error checking thus leading to an overall more stable program (see Section 4.5.3). In Open Content there is not necessarily a distinct method of determining an "error," and sometimes there is no definite right or wrong way to take the project. This leads to an interesting method of dealing with this. Many Open Content forums provide for communities to comment on a person's addition, or provide for a rating scale. Also, many Open Content projects have “moderators” much like Open Source that are in charge of screening through submitting material. These projects have nearly the same guarantee of quality as Open Source. The Open Content projects without these methods of assuring quality tend to be those without any aspect of centralization.

7 Current State of Open Content

Open Content as a cohesive movement is still in the infant stages of development compared to Open Source. However, there is a substantial base of examples from which to draw and a definitive structure to examine. Creative Commons, an organization dedicated to promoting the Open Content ideal, currently offers what is considered to be the standard Open Content licenses used to govern Open Content and provide a legal basis for it [18]. Creative Commons does include all of the commons, thus one of the licenses allows for limitations which overstep the bounds of Open Content. Each of these licences can be applied concurrently with other Creative Commons licenses. They can combine to create a legally protected form of Open Content.

There are four kinds of Creative Commons Licenses. The first license is an Attribution license. This license used alone allows users to essentially do anything to the content as long as the original creator is credited. The second license is a Noncommercial license. This license used alone states that a user can do anything to the content, but they cannot use that content or any derivative works for commercial purposes. The third license is a No Derivative Works license. This license used alone forbids the users to produce derivative works from the content. Obviously the use of the No Derivative Works license is not agreement with the Open Content movement (although the content would remain the commons). The last license is a Share Alike license. This license used alone requires any derivative works to be redistributed under the same license as the original content. Open Content as a movement usually has the Attribution, Noncommercial, and Share Alike licenses applied to the content. Sometimes the Noncommercial license is not used. The application of the Attribution and Share Alike licenses in tandem, however, meets the requirements we set forward for Open Content, and creates a licensing scheme analogous to many Open Source Software licenses (in particular the GPL, described in Section 3.1).

In order to truly analyze the success of the Open Content movement, we will examine two examples of Open Content in the “real world.” The first example will refereence Open Music as a subset of Open

Content and will be an example of the less centralized form of Open Content. The second example will reference Open Educational Works and will be an example of the more centralized form of Open Content.

7.1 Open Music

Open Music certainly falls under a category of the arts, and thus naturally this form of Open Content is less centralized than more functionally-oriented Open Content projects. It is difficult to find a common goal or vision for Open Music. It is hard for many people to work together on producing one song. Instead, Open Music has taken on more of a bulletin board format, where people create songs and make them available for people to download, modify, and distribute. In this example, we examine Magnatune.com, perhaps the most advanced Open Music website online.

7.1.1 Structure Summary

Although Magnatune has come a long way, it was opened not so long ago in May 2003. It was founded by John Buckman after watching his friends and wife suffer under traditional record deals. He created Magnatune to help the artists get exposure in a way that the public wants and to help encourage creativity among the public. An artist can simply send their music to Buckman and assuming Buckman finds the music to be of decent quality, it will be posted online. All the music is licensed by the Creative Commons Attribution, Noncommercial, and Share Alike licenses. This ensures that the public is free to enjoy the music, allows them to modify and distribute the music (gaining the musicians fans), but also prohibits the commercial use of the music without the artist's permission (guaranteeing just compensation). The music is offered at a reasonable price and apparently is catching on. The website now claims to bring in upwards of $15,000 a month off music sales. A quick check of the website reveals a number of press clippings mostly with positive reviews of the service [20]. As of yet, the majority of the music available on the website is from fringe groups with select musical taste, but it has the potential to grow. Similarly, the users who download Open Music from Magnatune tend to be interested in select genres of music, but also this has potential to grow.

There is an inherent limitation, however, in the structure of Magnatune. The only check for quality is the screening of John Buckman, who personally guarantees the quality of all songs available for download. There is no method for users to control the quality available, or even provide any publically available feedback for the music. Magnatune though, is still growing. The potential for expansion and the addition of new forms of quality checking is still present. A good idea for what form this quality checking should take could be taken from the comment system of a website such as Slashdot [24]. Slashdot is a news website that allows users to post their own news stories, but also to comment on stories. Overall, a system is put into place where a user of Slashdot can judge the quality of a post by the comments in response to that post. Magnatune could implement some sort of commenting system or rating system that would allow users to decide how good a musical piece is before wasting their time downloading something of poor quality. Perhaps pieces of poor enough quality would then be removed from the website. The filtering out of poor quality works, as occurs through the constant revision and peer review in Open Source programming, is a necessity for the establishment of a truly successful business model based on Open Content.

7.2 Open Educational Works

Open Educational Works, in contrast to Open Music, have a clear centralized goal and purpose. With MIT OpenCourseWare, a leading example of the genre, the goals are made accessible in a link off their website:

- Provide free, searchable, access to MIT's course materials for educators, students, and self-learners around the world.
- Create an efficient, standards-based model that other institutions may emulate to openly share and publish their own course materials. [22]

Similar to Magnatune, the content in MIT OpenCourseWare is licensed under the Attribution, Noncommercial, and Share Alike licenses. Also similar to Magnatune, there are some weaknesses OpenCourseWare's design. There are better quality control checks in this structure, but the quality control has limited the extent to which this website can truly be called Open Content. Only professors are allowed to add to the content available on MIT OpenCourseWare. This assures the quality of the material, but it limits the freedoms of the users. The professors, however, are able to add as much material as they choose on the topics of each of their courses, and one of the goals of OpenCourseWare is that other websites of similar structures will be created (perhaps allowing community interaction). Overall, the response to this website, despite the lack of community interaction, has been extremely positive.

Scrolling through the website reveals many stories and quotes by pleased users of OpenCourseWare from all over the world. Obviously the model, though slightly limited in openness, has been a success.

8 Conclusions: Is Open Content Viable?

We set out to discuss in this paper the viability of Open Content both as an information sharing model and as a business model. We examined whether Open Content provides sufficient motivation for contributors to create new works, for creators to publish and distribute those works, and for consumers to obtain these works. We have shown that the motivations for Open Content are in general the same motivations that have carried the Open Source movement to success. We have also shows that the motivations for creators are similar to that of Open Source. We have shown not only that users enjoy using this Open Content, but that it can provide a successful business in the form of Magnatune as an example. The one thing we have not shown is an example of a fully functional Open Content project with no flaws. We are not sure that such an example exists as of yet. There exist projects that meet all the requirements of Open Content, yet do not have the quality control necessary to carry a semi-successful business into a truly successful business. To fully reach its potential, the Open Content movement must a method of quality control and maintain a strict adherence to its principles. As indicated by the recent array of relatively mainstream media stories about the potentials of Open Content [11], [21], interest in alternatives to the current restrictive system is grown. There is no reason that Open Content should not in the near future form a viable and important alternative to proprietary content.

References

[1] Roy T. Fielding Audris Mockus and James D. Herbsleb. Two case studies of open source software development: Apache and mozilla. *ACM Trans. Softw. Eng. Methodol.*, 11(3):309–346, 2002.

[2] Yochai Benkler. Free as the air to common use: First amendment constraints on enclosure of the public domain. *N.Y.U.L.*, 74, 1999.

[3] Yochai Benkler. Coase's penguin, or, linux and the nature of the firm. *Yale Law Journal*, 111, 2002.

[4] Clint Boulton. Open source, proprietary code quality comparable. *internetnews.com*, July 2 2003.

[5] James Boyle. The second enclosure movement and the construction of the public domain. *Law and Contemporary Problems*, 66(1,2):33–74, 2003.

[6] Creative Commons. "Some rights reserved": Building a layer of reasonable copyright.

[7] Ben Crowell. Do open-source books work?, 2000.

[8] Joseph Feller and Brian Fitzgerald. A framework analysis of the open source software development paradigm. In *Proceedings of the twenty first international conference on Information systems*, pages 58–69. Association for Information Systems, 2000.

[9] Apache Software Foundation. How apache came to be. `http://httpd.apache.org/ABOUT_APACHE.html`.

[10] Various licenses and comments about them.

[11] Thomas Goetz. Open source everywhere. *Wired Magazine*, 11(11), November 2003.

[12] Intellectual property. *Microsoft(r) Encarta(r) Encyclopedia*, 1993-2003.

[13] Intellectual property for cs students.

[14] Stefan Koch. Effort, co-operation and co-ordination in an open source software project: Gnome. *Inform Syst J*, 12(1):27–27, 2002.

[15] Open Source Development Lab. About osdl. `http://www.osdl.org/about_osdl/`.

[16] Josh Lerner and Jean Tirole. Some simple economics of open source. *J Industrial Economics*, 50(2):197–197, 2002.

[17] Larry Lessig. The architecture of innovation. *Duke Law Journal*, 51, 2002.

[18] Licenses explained — creative commons. `http://creativecommons.org/learn/licenses/`.

[19] Lydia Pallas Loren. The purpose of copyright. *Open Space Quarterly*, 2(1), 2000.

[20] Magnatune. `http://www.magnatune.com`, 2003.

[21] Fiona Morgan. Copywrong. *The Independent Weekly*, December 6-9 2003.

[22] Massachusetts Institute of Technology. Mit opencourseware. `http://ocw.mit.edu/index.html`, 2003.

[23] Open directory project. `http://www.dmoz.org`, 2002.

[24] Inc. Open Source Development Network. Slashdot. `http://www.slashdot.com`, 2003.

[25] Christian Payne. On the security of open source software. *Inform Syst J*, 12(1):61–61, 2002.

[26] Bruce Perens. The open source definition. available at `http://www.opensource.org/osd.html`.

[27] Bryan Pfaffenberger. Why open content matters. *Linux Journal*, 2001.

[28] Associated Press. Brazil gives nod to open source. *Wired News*, November 16 2003.

[29] Eric Steven Raymond. *The Cathedral and the Bazaar*. Thyrsus Enterprises, 3.0 edition, 2000.

[30] Eric Steven Raymond. *Homesteading the Noosphere*. Thyrsus Enterprises, 3.0 edition, 2000.

[31] Eric Steven Raymond. *The Magic Cauldron*. Thyrsus Enterprises, 3.0 edition, 2000.

[32] Inc Red Hat. Red hat history. `http://www.redhat.com/mktg/rh10year/`.

[33] Gary Rivlin. Leader of the free world. *Wired Magazine*, 11(11), November 2003.

[34] Stephen Shankland. Study lauds open-source code quality. *CNET News.com*, February 19 2003.

[35] Sourceforge.net software map. `http://sourceforge.net/softwaremap/`, December 2003. information current as of 12/6/03.

[36] Peter N. Spotts. Who will build our digital future? *The Christian Science Monitor*, December 4 2003.

[37] Richard Stallman. Free software: Freedom and cooperation. `http://www.gnu.org/events/rms-nyu-2001-transcript.txt`, May 2001. transcript of speech given at New York University, New York.

[38] Richard Stallman. Misinterpreting copyright. 2003.

[39] David Sweet. Andamooka: Open support for open content. *Linux J.*, 2001(82es):13, 2001.

[40] Copyright law of the united states of america (Circular 92).

[41] Wikipedia. `http://en.wikipedia.com`, 2003.