Review of Economic Research on Copyright Issues, 2006, vol. 4(1), pp. 63-86

CAUSES, EFFECTS AND SOLUTIONS OF PIRACY IN THE COMPUTER SOFTWARE MARKET

AMY MARSHALL

Abstract. Much literature has been devoted to exploring the protection of computer programs. The decreasing effectiveness of copyright and patents has been extensively examined and alternative forms of protection, both physical and market-based, have been laid out. A large proportion of writings is dedicated to describing the significant network externalities that exist in the software market, and the effect that these have on the optimal level of protection. A large number of surveys have been undertaken to analyse the characteristics of software pirates and their incentives to pirate. This paper attempts to provide an overview of this literature.

1. Introduction

The widespread incidence of unauthorised use and copying of computer software has had a devastating impact on the software industry in recent years. Significantly costly time and effort is invested by software programmers in developing a program and with software piracy preventing these developers from recouping remuneration from users, the incentives existing to produce new software may not be sufficient enough for any further development, causing the industry to stagnate. This would be costly to not only computer programmers but also to the government who would miss out on a significant amount of taxation revenue and society as a whole, whose software choices would be diminished.

The traditional protector of intellectual property is copyright, defining the rights of ownership of creators and allowing for the enforcement of these rights. The basis behind copyright law is to provide the means for a creator to receive sufficient remuneration for development to occur. However, copyright must also provide a balance between this protection of the rights of the creator and the social gain which is earned from society's enjoyment of using the good. This is an especially relevant consideration in the case of public-good-type creations, and those with strong network effects, as the benefit to society of an additional user is high.[1] In addition, copyright must be careful to not provide overly strong protection, so as to make incremental useful innovations by subsequent creators too costly. Copyright also incurs two possible further costs, "rent seeking" by monopolists with inefficient allocation of resources and the transactions costs which are required to protect and

[1]The high cost to produce, low cost to use nature of intellectual property likens it to a public good. Efficient production would thus dictate producers receive a positive price whereas efficient distribution would mean consumers would pay a zero price. See Demsetz (1970) and Warren-Boulton et al. (1994).

enforce. A copyright regime must be careful to monitor and control these costs so that they do not outweigh the merits of the system.[2]

Copyright has had extensive experience in protecting books and journals from photocopying. However, distinctive characteristics of computer software, as laid out in Shy and Thisse (1999), have presented a very different subject matter to be protected. Unlike the diminishing quality of paper photocopies or copies of cassette tapes which loose colour and clarity, software may be repeatedly copied, from copies of copies, and the quality is identical to that of the original. This means that while the number of photocopies that can be made is dependent on the number of originals purchased, all software piracy could be copied from only one disk. It is difficult to physically protect publishers' rights against the illegal photocopying of journals and books, but software publishers are able to install protective devices making copying costly, difficult or even impossible. A further difference is that while reading copied books and articles or listening to copied music does not require reference to or operating instructions from the original publishers, users of software are dependent on developer's services and documentation.

In the face of these differences, much literature has been devoted to exploring the protection of computer programs. The decreasing effectiveness of copyright and patents has been extensively examined and alternative forms of protection, both physical and market-based, have been laid out. A large proportion of writings is dedicated to describing the significant network externalities that exist in the software market, and the effect that these have on the optimal level of protection. A large number of surveys have been undertaken to analyse the characteristics of software pirates and their incentives to pirate. This paper attempts to provide an overview of this literature.

The paper is set out as follows: firstly, a general description of the nature of the software market is presented. Section three provides a brief overview of copyright law as it applies to software. Section four describes the types of software piracy that occurs, including the increasingly threatening Internet piracy. Key international piracy rates and the financial consequences of piracy are presented. Section five follows, examining the extensive literature undertaking profiling software pirates. A brief analysis of motivating and cultural factors is given. Section six looks at the literature on network externalities and standardisation in the industry, and examines the possible profitability of allowing piracy that results. Section seven reviews literature outlining reasons why Copyright and Patents fail to protect software and offering possible alternative forms of protection. Section eight briefly overviews the phenomenon of open source, public domain, projects. Finally, section nine concludes.

2. The General Nature of Software

2.1. **Types of Software.** Several different categories of computer software programs exist, with different features and market characteristics. Hinduja (2003) classifies software into two groups; public domain software or shareware and freeware, and commercially-produced software. Individuals are permitted to copy and distribute an evaluation version of shareware programs for a given trial period. After this time, if the individual wishes to continue to use the program, he must

[2]For a good description on these costs and a general overview of copyright, see Plant (1934), Landes and Posner (1989), Koboldt (1995), and more recently, Liebowitz and Margolis (2005).

pay a registration fee. Individuals can copy, distribute, reverse-engineer, modify and freely develop derivative works of freeware on the condition that they remain designated as freeware, they are not allowed to be sold for commercial profit. These projects are further discussed in section eight of this paper. The most well-known product is commercially-produced software. To use this software, a licence for the program must be purchased from the manufacturer prior to its installation on a computer. It is with this type of software that the most common and serious software-theft occurs.

2.2. **The Software Market.** Richardson (1997) conducts a thorough examination of the nature of the software market. He identifies four key features of software necessary to understand how the industry works. These features are that using the technology incurs a zero marginal cost, that software features a rapid rate of innovation, that significant networks exist and that standards play an important role. The following is a brief overview of some of the Richardson's findings.

The costs involved when developing a new software programme are not contingent on the number of licenses sold for that product. It would seem that given these zero or near-zero marginal costs, the best result for consumers would be that the economies of scale are exploited with only one producer providing for the entire market. However, it could be that they would do better in benefiting from the economies of scale through lower prices, which would generally require that suppliers are faced with competition. Given these factors, it is thus difficult to determine what the ideal market structure for consumers would be.

A significant barrier to entry may exist in the software industry due to the presence of standardization. Introducing a new program may be difficult as consumers may be unwilling to switch due to the amount of application software available which is compatible with the existing standard products. To displace these, a new developer would be required to provide a set of new complementary products with additional advantages to compensate for the cost of a consumer switching. This barrier to entry would work to provide the manufacturers of well established systems with a degree of monopoly power. There are, however, possible weaknesses to this argument which Richardson points out. The new product developed may itself be compatible with the applications designed for the existing standard product. Further, it would not necessarily be a requirement that in order to compete with an established product, the new operating system must be able to run every existing application.

The second key feature that Richardson identified was that the software industry experiences a rapid rate of innovation. With this fast moving technology, the average life-span of a product is very short. A newly introduced product may be aggressively marketed to achieve high sales, which with zero marginal costs can allow both low prices and high returns. If the marketing is successful, the product may benefit from a large share of the market. However, a superior competing product may be being planned by another competitor and so in order to continue, the firm must itself be already planning its next product. This means that in addition to having concurrent competition in the software industry, there is also sequential competition. That is, while a product can dominate a market at a certain time, it is likely to be replaced by another product after only a short interval.

Richardson's final two features of the software industry regard the role of networks and standardisation. Industry standards emerge because consumers desire

their software products to be compatible. The number of users in a product's network increases the value of the product, that is, there is a positive network externality of software consumption. These issues are further discussed in section 6 of this paper.

3. Software Copyright Law

Software programs are intellectual property that is covered in the Copyright Law of the United States of America (and Related Laws contained in Title 17 of the United States Code) from their first point of creation with the aim of preventing misuse by others and to reward and encourage innovation (Hinduja 2003). This Law grants a copyright owner the exclusive right to reproduce the copyrighted work, make derivative copies based on that work and distribute copies of the work to the public. Section 117, 'Limitations on exclusive rights: Computer Programs', exceptions to the exclusive rights of copyright owners are listed. These include making an additional copy by the owner or adaptation by the owner of the copy, provided that creating the additional copy is essential in order to use the computer program, and it is only used for that function. Alternatively, the additional copy or adaptation can be used for archival purposes only and that once the computer program is no longer legally possessed, all archival copies and adaptations are destroyed. Section 117 also permits leasing, selling or otherwise transferring ownership of any legal additional copy, so long as all rights to the original computer program are also transferred. Any transfer of adaptations may only be done with permission from the copyright owner. Thirdly, owners of a machine are permitted to make a copy of a computer program for the purpose of maintenance or repair of the machine only if that copy is only produced in the process of activating a machine that contains an authorised copy of the program. Conditions for this exception are that the copy is only to be used for this purpose and once the maintenance or repair is completed the copy is destroyed (Circular 92 2003).

Further limitations on the exclusive rights of general copyright holders are outlined in sections 107 through 122 of the U.S. Copyright Act. For example, Fair Use permits reproductions of copies for the purposes of criticism, comments, news reporting, teaching, scholarship, or research. Factors used when determining whether a case is fair use include the purpose and character of the use and whether it is for commercial or for non-profit educational purposes, the nature of the copyrighted work, the proportion and significance of the copyrighted work used, and the effect which this use will have on the market or value of that work. Additional limitations in the Act include the reproduction by libraries and archives (with detailed restrictions) and use in certain performances and displays (Circular 92 2003).

The Software Rental Amendments Act of 1990 was passed by Congress in response to the frequent creation of pirated software copied when renting authorised original software. This Act deems it illegal to rent, lease or lend original copies of any software without permission of the original copyright owner (SIIA 2004)

The Business Software Alliance makes the Law regarding software use clear in their Fact Sheet "Software Piracy and the Law Software Piracy in the United States". According to this, by purchasing software, you are only purchasing the right to use software under certain restrictions imposed by the owner of the copyright which are described in the software's accompanying license. By breaking the terms of the license and using counterfeit software not only is one liable for legal

consequences, but also there is an increased risk of viruses or defective software, poor if any documentation, no warranties, little technical support available and you are ineligible for the software upgrades which are offered to licensed users.

4. General Information on Software Piracy

4.1. **Types of Piracy.** Software Piracy can be broken up into several categories, as described by Crampes and Laffont (2002). *Softlifting* is when one licensed copy of software is purchased and is loaded onto multiple computers, against the conditions of its licence. Sharing software with friends and co-workers is softlifting. The illegal duplication and sale of software, in a way intended for it to look legitimate, is known as software *counterfeiting*. *Hard disk loading* is when unauthorized copies of software are loaded onto computer hardware by computer sellers to make the purchase of the hardware from the seller more attractive. On the other hand, *unbundling* is when software which was originally intended to only be sold packaged with specific complementary hardware is unbundled and sold on its own. *Renting* is when unauthorized software is made available to temporarily hire. Finally, *Internet piracy* is illegally providing on the internet, or downloading from the internet, software itself or the means to violate copyright protection mechanisms, such as serial numbers and cracker utilities.

4.2. **The Business Software Alliance.** The Business Software Alliance (BSA) is an organisation, formed in 1988, whose goal is to stamp out international software piracy. Its strategies include education and lobbying for effective copyright laws in countries that lack protection. Litigation, they claim, is only one of several antipiracy strategies (Gwynne 1992). In their December 2005 Report, "Expanding the Frontiers of Our Digital Future", the BSA describe five key findings prompting the need to reduce piracy. Lower software piracy rates would produce higher IT benefits; a decrease could generate faster IT growth, from 33 to 45 percent growth between 2004 and 2009 with a 10-point reduction in software piracy over that period; this faster IT growth could increase Global Economic Output; those countries who suffer from the highest piracy rates would enjoy the most benefits from decreased piracy; and every region would benefit from a decrease in software piracy.

In its report the BSA identifies the parties who stand to benefit from this 10-point reduction in software piracy over four years. Consumers would be able to enjoy greater competition and increased choices. An estimated 2.4 million new jobs would be available for workers. The software innovators would be able to be compensated financially for their creativeness. There would be new opportunities to market, sell, distribute and service software, which will benefit entrepreneurs. Finally, governments would have the benefit of an additional $67 billion in tax revenues. This money could provide 33 million computers for schools, health care for 45 million, college educations for 6.6 million, schooling for 11 million children, job training for 435 million, or 132 million families with such services as day care, maternity, or home help.

In their December 2005 Report, the Alliance outlines five concrete steps for reducing software piracy. They seek to implement the World Intellectual Property Organization Copyright Treaty. This treaty was adopted in 1996 in response to increasing threat of Internet piracy and is aimed at enabling more effective control of digital and online piracy through improvements in copyright laws. Secondly, the

Alliance wants governments to increase their enforcement mechanisms, as required by the World Trade Organization's Trade-Related Aspects of Intellectual Property Rights Agreement. The Alliance then wants to target resources more effectively by creating intellectual property enforcement units at the national level, increasing cooperation between police and other enforcement agencies by supporting the training of law enforcement and legal officials and by providing the required tools to fight piracy to those requiring them. This will help increase the significance of punishment, and thus act as a better deterrent. By increasing public education and awareness the Alliance hopes to reduce piracy as this will cause shift in the public's attitude which is often a requirement. This can include teaching the consequences of breaking the law and emphasising the government's intention to strictly enforce copyright laws. Finally in their five step plan, the Business Software Alliance wants to lead by example with governments, the world's largest software users, showing they will not tolerate piracy in the public sector and have strict controls on their software assets. This will demonstrate the government's commitment to fighting piracy in the private sector.

4.3. **Internet Piracy.** The growth of the size and popularity of the Internet had an overwhelming effect on software piracy and it is regarded as possibly the fastest growing type of software piracy (BSA n.d. b). The Internet faces the same laws and license agreements as physical software distribution; no differentiation is made between offline and online infringement in the U.S. Copyright Act (BSA n.d. b). Warez Web sites, allowing customers to choose as much pirated software as can fit on a CD which is then delivered right to the customer for a minimal fee, are numerous (Gopal and Sanders 2000). Internet Auction sites often appear to be legitimate and can thus catch low-budget consumers, some of whom would not purchase illegal software knowingly. A technique used to convince consumers include labelling the illegal software as an authorized back-up copy or claiming the product is genuine and was cheaply obtained wholesale or through liquidation or bankruptcy sales. Subsequent purchase testing has revealed that almost always, these claims are untrue (BSA n.d. b).

The Business Software Alliance (BSA n.d. b) describes three main factors that are leading to this rising threat. Firstly, because the number of users of the Internet has grown so large in both the home and office environment, the Internet is now a vast marketplace for many goods, including pirated software, which is without limit and always operating. Secondly, the barriers to entry to the Internet that once existed, such as bandwidth, cost and the requirement of extensive computer knowledge, are now lessened or removed meaning that accessibility to the Internet has greatly increased. Finally there is currently a lower risk of detection for Internet piracy than many of the other forms of illegal circulation. The BSA has developed a plan to combat this growing Internet piracy, including deploying an Internet Crawler system. This identifies illegal files that are offered on Internet sites. The Alliance engages in high profile and impact civil litigation and it encourages the criminal prosecution of Internet pirates by making sure the law has good coverage, developing and supporting criminal cases, and offering training to enforcement authorities.

4.4. **Facts and Figures.** The Software Publishers Association (SPA) estimates software piracy by comparing the number of computers sold and the number of

software applications sold. The difference between total number of applications expected to be sold calculated from the average number of applications used per computer and the number of computers sold, and the number of applications which were actually purchased gives an approximation of the number of software applications that were pirated (Gabella and Picasso 1995).

Software piracy is a problem to world economies due to the immense size of the IT industry and its heavy influence on economic growth. The sector is made up of 1.1 million businesses, with 11 million high-income employees. This generates almost $900 billion in taxes per year, which in turn contributes $1.6 trillion dollars to the global economy annually. The BSA estimates that throughout the world, between 2004 and 2009, over $1 trillion dollars will be spent by consumers and businesses on software, that is around one out of every five dollars on IT. (Findings from an IDC analysis assessing the IT sector's economic impact in 70 countries around the world and the benefits that can accrue to countries that reduce software piracy, BSA 2005). Software is therefore an important driver in increasing the benefits of the IT sector. Software-type jobs and tax revenues will increase by at least five percent annually between 2004 and 2009 (BSA 2005). The industry is growing rapidly.

Software piracy is the IT sector's most serious problem (Anthes 1993). In their December 2005 report, the BSA estimates that for every two dollars that is spent on purchasing software legal, one dollar's worth is obtained illegally. The global piracy rate is currently around 35% but rates vary greatly between individual countries, with the US enjoying the lowest rate of 21%, but countries such as Vietnam, China and Indonesia suffering from high piracy rates of 92%, 90% and 87% respectively in 2004 (BSA 2005).These sorts of countries are often called "one-copy-countries" meaning that there is only one legal copy that serves a large proportion of the whole market (SIIA 2000). Generally there is an inverse relationship between the size of a country's IT sector as a percentage of GDP and its software piracy rate. This means that a country such as the US, with a low piracy rate, still suffers greatly from piracy. In fact, because the US has the largest software market in the world, worth $7.2 billion, they also lose the most revenue from software piracy, $1.8 billion in 2001. Japan and China follow closely behind. The Asia-Pacific region suffered the most of any region with a loss totaling $4.7 billion (2001). Western Europe follows with a loss of $2.7 billion. In total, the amount of revenue estimated to have been lost world wide to software piracy is $11 billion (Rasmussen 2003).

5. Profile of Pirates

Many empirical studies have been undertaken evaluating the characteristics of software pirates and what influences them to illegally copy. Most of these involve surveying college students, identifying if they pirate, how often and what type of program, and what are their motivations for this action (see, for example, Im and Koen 1990, Sacco and Zureik 1990, Rahim, Seyal and Rahman 1999, Chiang and Assane 2002, and Hindujua 2003). Understanding characteristics and motivations of pirates is important in the absence of effective antipiracy law in order to address the software piracy problem (Cheng, Sims and Teegen 1997).

5.1. **Characteristics of Pirates.** College students are the most frequent software pirates, which is why a large proportion of the literature is dedicated to surveying them. They possess several characteristics which make them liable such as their high demand for programs, their low level of disposable income and their

knowledge and ability to pirate. However, these characteristics can be somewhat negated by software companies offering student discounts, the need for students to purchase costly CD writers to copy (although this requirement is diminishing), the well equipped computer labs offered by universities and the possibility that students could be more knowledgeable about copyright laws. (Chiang and Assane 2002). Many studies describe the typical software pirate as a young male who is experienced with computers, such as students majoring in computing (Solomon and O'Brien 1990, Simpson, Banerjee and Simpson 1994, Wood and Glass 1995, Sims, Cheng and Teegen 1996 and Rahim, Seyal and Rahman 1999, Chiang and Assane 2002). Hindujua (2003) provides further characteristics of student pirates such as income of parents and the location and type of educational institution. Some authors however, (for example Harrington 1989, Sacco and Zureik 1990) claim that it is situational variables and not these personal characteristics that are most important in explaining software piracy.

The effect of morality on the tendency to pirate is analysed by several authors. Logsdon et al. (1994) hypothesized that individuals with higher levels of moral judgement will be less likely to approve or commit illegal copying. They found that their data was unable to strongly support this hypothesis. The issue of piracy is perceived with low moral intensity; the act was not perceived to be good, but it was definitely not perceived to be evil. This finding is in agreement with that of Simpson, Banerjee and Simpson (1994). Logsdon et al. conclude that it is therefore important for software companies to concentrate on increasing awareness about the nature and size of the harm caused by pirating so as to increase its moral intensity. Subjective norms, peer beliefs and the frequency of other's infringements have also been examined and been found to have a significant influence amongst college students on the propensity to pirate (Sacco and Zureik 1990, Peace, Galletta and Thong 2003).

Im and Van Epps (1991) find that the type organization has a large effect on the amount of software piracy that occurs. Piracy is more prevalent in the academic environment, largely due to less enforcement of copyright law by academic officials. In addition, controversy surrounds the "fair use" doctrine in the academic community with many believing educational institutions are permitted to make copies of software for educational purposes. However, this "fair use" exception was not created with the intention of permitting unlimited copies by educational institutions. Im and Van Epps also note that more piracy is reported by managers in the not-for-profit, public sector and consulting industries than service and manufacturing managers. Reasons for these differences could include that industries working for profit are better able to afford software and boast a more controlled and supervised environment, which makes the law and penalties for copyright violations clear to employees.

5.2. **Motivating Factors.** Unanimously in the literature, price is agreed to be the key motivating factor for people, especially students, to pirate software (Im and Van Epps 1991, Chen and Png 1999, Hindujua 2003, Chiang and Assane 2002). There appears to be a strong income effect with household income heavily influencing the pirate-purchase decision (Cheng, Sims and Teegen 1997, Gopal and Sanders 2000). Other factors influencing piracy are the lengthy time required to purchase software legally (Im and Van Epps 1991, Simpson, Banerjee and Simpson 1994), not knowing where or how to purchase legal software (Simpson, Banerjee and Simpson 1994),

the monitoring strategy of the software publisher (Chen and Png 1999) and even the personal challenge of pirating (Simpson, Banerjee and Simpson 1994). Software companies seem remote and intangible on the Internet, the victims of piracy are more difficult for individuals to identify, reducing the harm they believe they commit by pirating (Logsdon et al. 1994, Hindujua 2003). Hindujua also comments that individuals blame software corporations for being greedy and profiteering and that by failing to use sufficient copy protection schemes, manufacturers deserve to have their products pirated.

Cheng, Sims and Teegen (1997) examine what factors would influence an individual to choose to purchase, as opposed to pirate, a piece of software. They found that software was more likely to be purchased if it was required for the school or workplace, likely to be used very often and if it included a useful manual. The increasing inclusion of in built help services in programs has reduced this benefit of purchasing software.

The impact of knowledge of copyright law has been debated in the literature. Because few have a full comprehension of copyright laws and, many believe that these laws are never enforced Christensen and Eining (1991) claims that the impact is not significant. In their study of college students, however, Chiang and Assane (2002) find that students who have a greater knowledge of copyright laws believe they have a higher risk of being caught. They then found that there was a high correlation between infringement and a student's attitude towards the risk of being caught and convicted. Lodgson et al. (1994) found only a slightly positive relationship between legal knowledge and an individual's attitude towards software copying, concluding that knowledge was not a significant deterrent of the common acceptance of copying.

5.3. **Cultural Factors.** Various studies have been undertaken analysing piracy in different cultures, especially in Asia. They examine the differences in moral beliefs and other factors which contribute to the large variance in piracy rates. Asia is swiftly developing into the software and electronics capital of the world. This means that software piracy in this area is becoming especially alarming for the industry (Gwynne 1992). Outlets exist such as Hong Kong's Golden Arcade, Singapore's Funan Center and People's Park, and Taiperi's Computer Alley, where shoppers can purchase pirated copies of almost any software available for not much more than the cost of a blank disk. A measure of the total illegal sales from these locations is impossible to calculate (Swinyard, Rinne and Keng Kau 1990). Malls selling legitimate software such as the Asia Computer Plaza find it extremely difficult to compete with these shops, despite offering heavily discounted prices (Hebditch 1986). Illegal shopping arcades are deeply rooted in the Asian market.

Swinyard, Rinne and Keng Kau (1990) sought to examine differences in Asian piracy rates by analysing data on morals and behaviour in Singapore versus the United States. Acknowledging that Asian cultures usually emphasize that individual creators or developers should share their developments with society[3] and that Asians tend to be circumstance-oriented in decision making as opposed to America's rule oriented approach, they hypothesised that Americans' attitudes and

[3]See also Marron and Steel (2000) who discuss the collective nature of non-Western cultures and its influence on intellectual property protection and software piracy rates. This study is then extended by van Kranenburg and Hogenbirk (2005) who examine cross-national variation in piracy rates in four US copyright-based industries.

intentions would be more in line with copyright laws than that of Asians and that Asians would tend to base decisions on the outcomes of their behaviour, whereas Americans would base their decisions on the nature of the decision itself. These beliefs were supported by data and the authors concluded that because protection legislation such as copyright did not fit within the sharing culture of Asia, it would be incorrect to believe that copyright legislation would be supported by Asians and that a decrease in piracy could occur without a change in Asian culture norms.

Swinyard et al.'s study is complemented by studies on piracy by students in other countries such as Hong Kong (Wong, Kong and Ngai 1990), Saudi Arabia (Al-Jabdi and Abdul-Gader 1997), Brunei Darussalen (Rahim, Seyal and Rahman 1999), and Thailand (Kini, Ramakrishna and Vijayaraman 2004). Kini et al. compared levels of moral intensity of Thai to US students. They found that US students had a higher level of moral intensity which could contribute to the significantly higher piracy rate in Thailand. It was also found that students were heavily influenced by the level of moral intensity of their immediate community. This finding is supported by Al-Jaddi and Abdul-Gader (1997) who found that peer beliefs have a strong influence on the tendency to observe or engage in software piracy for Saudi Arabian students.

A major problem in Asia is a lack of knowledge about copyright laws. In several countries copyright protection is a new concept and many small Asian firms may not even realize that by copying software, they are breaking the law. For this reason, the Business Software Alliance has engaged in running seminars for local businesses which include specific information about the local piracy laws. Litigation in these countries is often difficult. Even when copyright laws in the area exist, the Alliance still has to convince local law firms and enforcement authorities to cooperate. In some countries, the Alliance does not even attempt to take legal action (Gwynne 1992).

While income levels certainly affect consumers' ability to pay for software, the Software Publishers Association is doubtful that per capita income is an important issue, claiming that an individual in a developing country who can afford to buy a computer must be affluent enough to be able to afford software. However, countries with higher piracy rates do tend to be skewed towards those with lower per capita incomes (Gopal and Sanders 2000). Gopal and Sanders believe that one reason for high piracy rates in developing countries is the high price of legal software. For example, one could purchase Microsoft Windows95 for $255 in Mexico in 1998, whereas in the US it would have only cost $110. Prices are set at United States levels, which are higher than most individuals in other countries can afford. They therefore believe that in order to combat international software piracy, the pricing issue needs to be addressed, with price discrimination[4] offering a viable way for the industry to reduce global piracy rates. A possibility is to index prices to the per-capita GNP of a country, with arbitrage between countries being regarded as an illegal transaction and treated accordingly. Gopal and Sanders believe that treating developing countries in this way will increase the willingness of governments to enforce intellectual property rights in order to enlarge their own domestic software industry and tax revenues from legal software sales. This strategy can dramatically increase revenues for software publishers.

[4]Forms of price discrimination in the software industry already exist such as publishers offering licenses as a form of bulk discount and the discounted prices offered to students (Gopal and Sanders 2000).

6. Network Externalities, Standardisation and the Possible Profitability of Piracy

6.1. **Network Externalities.** A key feature of the software industry is the presence of network externalities. These exist when the value of a product is influenced by the number of users of the same product (David 1985; Katz and Shapiro 1985). In the computer industry, as the user-base of a program increases in size, a consumer's willingness to pay for that program also increases. The literature is extensive in its analysis of the size of these network effects and their influence on the standardization of software products and the desirability of piracy (Several papers outline the possible profitability for software firms in allowing or even promoting piracy, in the presence of strong network effects (see, for example, Takeyama 1994, Slive and Bernhardt 1998, and Poddar 2002).

There are several reasons for the existence of network effects in the software industry. Primarily, users desire the ability to transfer files from their system to the system of others and between different formats within their own system (Gandal 1995, Katz and Shapiro 1999). It is necessary that the software each agent is using be compatible for this transfer to take place, in order that data and other files can be successfully read. A leading example is the success of Lotus Software's spreadsheet application, Lotus 1-2-3. Network externalities lead to this program becoming the principal package in the market. The ease with which agents could both read and write data in the program's file format placed the program at the centre of the network, facilitating straightforward transfers of numeric data for databases, spreadsheets, accounting and statistical programs (Slive and Bernhardt 1998). As the popularity and thus the size of the user-base of Lotus 1-2-3 grew, these network externalities increased the value of the program to consumers, cementing it as the industry standard in spreadsheets throughout the 1980s and early 1990s (Brynjolfsson and Kemerer 1996).[5]

A second factor contributing to network externalities is the heavy investment of learning time in order for a software package to be usable. Only a small fraction of the total cost incurred by an agent when acquiring a new software program is the paid purchase price (Brynjolfsson and Kemerer 1996). While the complex nature of software makes these additional learning costs high, an increase in the number of users of the software brings these costs down, with consumers sharing their knowledge of how to use a program. When a program enjoys increased consumption by home users, that package becomes more attractive for businesses. Employers can utilise their employees' existing knowledge of the program, a knowledge which the employees acquired at their own, and not the company's, expense. With no need for additional employee training, the total cost for the business of installing the program is reduced, thus making that program more desirable (Slive and Bernhardt 1998). In addition the ability for employees to bring their work home, given the same software is available at both their home and work office, may be attractive for firms. This will again increase the program's value (Slive and Bernhardt 1998).

The costs associated with learning a new piece of software can also be reduced through the provision of learning tools such as "how to" handbooks and tutorial programs by either the software developer or an independent party. The supply of these complementary products is directly proportional to the number of users

[5]See Lewis (1991) for a description of standardization among software user-interfaces.

of that software (Conner and Rumelt 1991, Church and Gandal 1992). It may also be profitable for a software developer to supply additional compatible add-ons to the program given a sufficient market size (Takeyama 1994). Both of these possibilities contribute to the network effect, making a more widely used software package more valuable to a potential purchaser. This is similar to the purchaser of computer hardware who considers how many users of similar hardware exist as this will heavily influence the quantity of software programs that are developed for use with a given computer (Katz and Shapiro 1985).

6.2. **Standardisation.** Standards in software products not only emerge not only due to the success of an individual product such as Lotus 1-2-3, firms may be willing to work together to establish compatibility standards when it is profitable for them to do so, such as when demand is highly contingent on their products degree of inter-operability (Richardson 1997).

Although standardisation may seem desirable in allowing the easy transfer of data and knowledge for consumers, it presence is not without problems. Standardization can be detrimental to the software industry as it can reduce competition and therefore consumer choice by eliminating the incentive to innovate (Barret 1993). A product which has reached the status of industry standard may be so deep-rooted in the market that it earns undue profit for its licenser. It would not be possible, however, to bring such a product into the public domain unless the public is willing and able to meet the cost of developing and maintaining these standards. Another possibility is restricting the profits earned by either limiting the life of the intellectual property protection or by controlling the amount charged for licenses. It is difficult to determine whether this approach would be in the public's best interest. The nature of the software industry, with the chance of a vast success being counter-balanced by prospects of overwhelming failure, makes it hard to define and control the notion of "undue profits" (Richardson 1997).

6.3. **Potential for Profitability.**

6.3.1. *Diffusion.* The increase in the value of a software program resulting from network externalities can lead to an increase in a firm's profits, even when the program is heavily pirated. The existence and extent of network externalities is not contingent on whether or not the user of a software program was a legal purchaser or a pirate. A pirate knowledgeable in the workings of a program is equally as beneficial as a legitimate purchaser in spreading his knowledge, reducing legal users' costs of adoption and thus encouraging others to purchase the program (Slive and Bernhardt 1998). The supplier of independent complementary products to the software cares only what the overall size of his market is, not whether the software has been pirated, when making his decision regarding the profitability of producing these goods and services (Conner and Rumelt 1991).

In their 1995 paper, Givon, Majajan and Muller propose a diffusion model approach to examine pirate's influence on software diffusion. This approach is especially relevant for software which relies heavily on word-of-mouth discussions between potential and existing users to increase the user-base. Pirates are thereby important in influencing potential users to use, and often legally purchase, the software. This is evident with the success of Lotus 1-2-3, which was very heavily pirated during its successful era (Slive and Bernhardt 1998). They consider the possibility that there may in fact be some difference between the influences on potential users

of legal buyers versus pirates. People may find it psychologically easier to copy from a pirate; buyers may be more credible having themselves invested money in the software; and, due to this investment, legal buyers are more likely to heavily use the software, and so may be more likely to influence others. Givon et al. go on to note that because pirates generally do not advertise themselves to be pirates, there is no reason to expect that their influence is any less than legal software purchasers.

This influence on the market by pirates is known as shadow diffusion and is more difficult to measure than legal diffusion which can be estimated with sales data (Givon et al. 1995). The authors' recommendations are that instead of seeking to eliminate shadow diffusion, firms should try to convert it into legal diffusion through various marketing methods. Punitive mechanisms such as protection strategies and lawsuits can act to destroy shadow diffusion, which can be detrimental to the firm. Givon et al. argue that due to the immense impact of pirates on buyers, any action by the software firm is best attempted at a later stage, having taken advantage of the influence of pirates. They also comment that a very large pirate group may also provide assistance to the firm as a barrier to entry for a potential competitor attempting to introduce a new product into the market.

6.3.2. *Price Discrimination.* Another angle which can be taken when viewing the positive aspects of piracy on profits is that of its ability to act as a form of price discrimination. There is a substantial difference in the willingness to pay of consumer groups for software products. Home users are likely to have a lower willingness to pay, partly because it is difficult to catch and punish consumers. Business consumers on the other hand are easier to monitor and thus have a higher chance of being caught, greater penalties can be extracted from them if they are caught, and the network externalities are likely to be more beneficial for business consumers. Thus this group is likely to have a higher willingness to pay (Slive and Bernhardt 1998).

Software producers would therefore find it advantageous to be able to price discriminate between home and business users. This would act to increase the number of users and thus the positive network effects (Takeyama 1994). If the enforcement costs of such price discrimination were not restrictively costly, it would then be best to heavily invest in piracy prevention. However, due to the difficulty in segregating types of consumers, it is difficult to price discriminate in the software market. Firms can then consider piracy as an alternative method of price discrimination, with pirated software enjoying a price of zero (Slive and Bernhardt 1998). Consumers with a low valuation of the product copy the product, at no cost to the firm, whilst consumers with a higher valuation pay a higher price for the product, enjoying the benefits of a larger network. In order to achieve a similarly large network, a software manufacturer would need to lower the price on all units (Takeyama 1994).

Consumers can also be grouped into early adopters and late adopters, again each with a different willingness to pay for software products. The increased risk for early adopters including heavy investment in learning time for an unproven product and a smaller user-base reducing the size of network effects would mean they are likely to value the product less than those who adopt the product at a later stage when the product has proven itself in the market and enjoys a large number of users. It is possible for a firm to reward and encourage early adopters by price discriminating intertemporally, charging less for the product in its introductory stages and increasing product with the increase in the network size and thus the

value of the product to consumers. An alternative is to allow early adopters to pirate the product and increase its value. If this occurred we would expect to see anti-piracy spending increase as the popularity of the product increases. This is commonly observed in reality, with successful and well established firms incurring the majority of anti-piracy spending. New firms to the market have more to gain or less to lose from allowing piracy (Slive and Bernhardt 1998).

A further way in which a software manufacturer can increase their user base is to give away free software to such places as universities and retailers, who will further increase the user base. Allowing piracy is an effective method of gift-giving in that it ensures that the recipient of the gift is someone who actually wants it and all the costs of gift-giving are borne by the recipient and not the firm (Conner and Rumelt 1991).

The ability of piracy to act as price discrimination and increase the user-base of a software product and thus the size of network effects and the value of the product means that it may be optimal for a manufacturer to tolerate some piracy by low-valuing consumers, typically home consumers, in order to increase demand by consumers who are then willing to pay more for the product, typically business consumers (Slive and Bernhardt 1998). Not only is it possible for firm profits to increase, social welfare can also enjoy a Pareto improvement. This suggests that the usual measures can overstate the harm of piracy to society (Takeyama 1994).

6.3.3. *Criticisms.* Poddar (2002) disagrees that increased profits are the general result from allowing limited piracy by some consumers. When this is shown to be the case, he claims the result is simply an artifact of the model and is therefore unsuitable as a universal explanation of the existence of software piracy. He then develops a model of his own which shows that even in the face of very strong network effects, it is always optimal for a manufacturer to protect from piracy, and that the incentive to protect is even higher than with lower levels of network externalities. This is due to the dampening effect on price resulting from competition that occurs when a pirate enters. Banerjee (2003) agrees with this result, remarking that with network externalities, the incentive to monitor and prevent piracy is higher. Poddar also comments that the ability for piracy to act as a form of price discrimination is highly dependent on the nature of competition when there is a duopolistic or any strategic market structure.

6.4. **Detection and Measurement of Network Externalities.** Firms have been gradually removing protection against copying since personal computers were introduced. This is partly because users find the protective devices annoying and detrimental to the efficiency of the product and partly because of the benefit increased users have on the value of the product (Shy and Thisse 1999). Not only is enforcement and prevention spending by the industry lower the optimal level that would be implied if firms were completely against piracy, but by providing freely downloadable evaluation versions of software products, manufacturers are giving their silent approval of piracy. Little effort is made to catch individual pirates; allowing limited piracy is optimal due to the presence of network externalities (Slive and Bernhardt 1998).

Another approach to examining the extent of network externalities is to identify features which are included to preserve compatibility. These can include the file format chosen and the design of the user interface. An empirical investigation

can then be undertaken, analysing the relationship between the size of a product's network and its compatibility or lack-there-of with the dominant standard on the product's price (Brynjolfsson and Kemerer 1996). For example, Gandal (1994) found that when a spreadsheet is compatible with the Lotus system or has links to external databases, consumers have a higher willingness to pay. This again indicates the presence of network externalities in the software industry.

7. Other Protection Strategies

7.1. **Why Patent and Copyright Fail to Work.** Dealing with multimedia is not a new problem for intellectual property protection, for instance films are multimedia (Cohen 1993), but computer software programs have caused new problems for the existing protection regime. Traditionally, protection using Copyright has been favoured as it is internationally recognised as being essential for computer programs given the widespread copying and transfer of programs between nations (Harkin 1993). However, programs are classified as both a "literary work" and a "machine" which causes complications when attempting to apply traditional Patent and Copyright laws to protect software. In addition, Trade secrecy laws are unable to offer protection for marketed software products because most of the knowledge required to recreate programs is located on the surface of the product, making replication a simple task (Samuelson et al. 1994).

Patent law is clear in excluding "printed matter", including methods embodied in texts, from is domain. This meant that for nearly two decades applications for patents for software innovations on subject matter grounds were rejected by the Patent Office. However, since the mid-1980s, seeing a need for increased legal protection of programs than that provided by Copyright, a wide range of software products have successfully been granted patents. A major issue is that patents usually protect methods of achieving a result, not the result itself. So while a patent for one method can be held, this cannot prevent software being developed which finds the same result in another way. This is problematic because often the results are the most valuable part of the program. In addition, to be eligible for a patent, significant advance over previous works is required and software development is largely incremental. Protection of software with patents would thus fail the economic goals of the system which are to grant rights only when a significant contribution is made (Samuelson et al. 1994).

Copyright law protects the expression of ideas, not machines and technological processes. It does not match well to software largely because it does not concentrate on the useful behaviour of the program, which is its main source of value. Protecting program texts does not prevent someone from copying behaviour, as these are mainly independent. However when program behaviour is expressive, it is able to be protected by copyright law. Courts have had some difficulty in applying traditional copyright infringement tests to software which has led them to develop new, sometimes inconsistent, tests for reviewing cases. With no prior experience in regulating technological fields, copyright has no obvious answer to many questions asked in software cases (Samuelson et al. 1994).

7.2. **Preventive vs. Deterrent Controls.** When deciding how best to protect the content of software programs, developers have the choice between deterrent and preventive controls. A preventive control is a form of technological protection which aims to increase the costs of pirating. These controls can include only providing

support to registered customers, including documentation that is difficult to replicate (Gopal and Sanders 1997) and embedding special codes in software to increase the difficulty of copying (Morgan and Ruskell 1987). On the other hand, deterrent controls do not directly increase the cost of piracy. Rather, they seek to deter copying through the threat of punishment. To achieve this, information about the illegality of pirating software and the effect of piracy on software development is distributed to the public and then backed up by investigations and legal campaigns aiming to extend and enforce copyright laws (Gopal and Sanders 1997). Gopal and Sanders (1997, 2000) examine the benefits of each form of control and find that deterrent controls have the potential to increase profits whilst preventive controls will generally decrease profits. This is possibly the reason why so few publishers utilise preventive controls in their protection strategy.

7.3. **Market Oriented Approach for Legal Protection.** In response to the apparent failure of copyrights and patents to protect computer software programs, in their 1994 paper, "A Manifesto Concerning the Legal Protection of Computer Programs", Samuelson et al. propose a two part strategy involving firstly a protection scheme which is organised around software's sources of value, that is its behaviour and the know-how that produces it, and secondly, a scheme which is grounded in principles of market economics and market preservation. The following is an overview of Samuelson et al.'s main findings and suggestions.

7.3.1. *Basis of Market-Oriented Approach.* A market oriented approach is a legal regime which would only protect the know-how in programs enough to avoid market failure and restore the competition which occurs when innovators have sufficient natural lead time in product development. Generally, competition is seen as healthy, increasing social welfare. Allowing reverse engineering and incremental innovation significantly contributes to total innovation with useful improvement being made on the original product. Given sufficient lead time, even with such competition firms should be able to receive adequate returns to provide enough incentive for further innovation. However, because much information in software products is stored near the surface, this required natural lead time may not exist. Therefore the rule of free competition needs to be modified for these products. This is especially true as the software market continues to develop, with emerging barriers such as increased costs of switching for consumers, meaning that stronger innovations are required to capture a large market share. This type of innovation requires longer design and development periods which further increase the problem of insufficient lead time.

The market-oriented approach may differ from a traditional intellectual property approach in that instead of granting a set of exclusive property rights to creators for a set period of time, programs could employ "blocking periods" during which only certain uses of the product would be prohibited. These periods would be able to be narrowly designed so as to only provide the amount of artificial lead time required to prevent market failure. For example, a longer blocking period could provide against reusing an innovation by developers in the same market compared to those wanting to reuse the same innovation in a remote market. There are other techniques the approach could utilise, such as requiring that users acquire a royalty-bearing license for certain uses. This would compensate developers for second uses of their innovation. Again, these licenses could take into account which

market segment the second comer operated in, as well as the nature of it use and the similarity between the two goods.

For this market-oriented regime to be successful, criteria need to be developed to assess when market failure is likely to occur. Samuelson et al. recommended considering three primary factors: the size and nature of that being imitated; how the second-comer obtained access to the entity and his creation's dependence or independence on the original; and how similar the two products and markets are. Obviously, there is an increased risk of market failure if the second comer's development is fast, easy and highly dependent on and similar to the first product and the two products exist in close markets.

7.3.2. *Goals and Principles of Approach.* In their manifesto, Samuelson et al. provide a list of the goals and principles which they would hope to achieve with the development of the market-system. Firstly, the system should build on existing legal foundations, not start afresh. While theoretically a completely new regime which protects all aspects of software may seem ideal, the disruption this would cause deems this to be unpractical. The most serious problems should be focused on, with difficult questions proposed by small cases best avoided. The regime should be predictable in its scope and duration of protection, in order to encourage investment and reduce the likelihood of litigation, an action the regime should discourage. Any legal distinctions made need to be technically coherent so that legal questions raised can be meaningfully answered by technical witnesses and experts.

It is important that the regime be able to evolve as technology evolves and it should not block access to know-how, only regulate its use, thus encouraging innovations that enhance products. A further way for the regime to encourage useful innovation is to avoid market failures which discourage investment. This is done by providing a reasonable lead time to investors allowing innovators to recoup their research and development costs plus earn some return of their investment, providing sufficient incentives for development but only so long as it is valuable. Reimbursing effort or expenditure alone can lead to inefficiency. Research and development costs should be shared by participants in the market, which can work towards avoiding the reduplication of effort required with trade secrecy to reach the same know-how independently. This could be achieved by requiring firms with little substantial research and development to either contribute to the costs of firms whose products they wish to appropriate, or to refrain from appropriating the product for a set blocking period.

Further goals of Samuelson et al. include the attempt to be able to distinguish among different types of second-comers. A number of factors looking at the market effects of their borrowing would determine the fee payable or whether the user would be blocked for some period of time. The regime should be "self-executing", that is it should provide protection with minimal amounts of administration and time expenses, to make the cost of obtaining protection low and thus make the regime for market friendly. Care must also be taken that while protecting against market failures, artificial barriers to entry do not emerge which could stifle innovation. Finally, the ultimate goal, they believe, is not just to prevent market failure, but to use this as a means to increase consumer welfare. Incentives to innovate will not only benefit developers but will also add to society's wellbeing with an increased of desirable products available. Overprotection of program innovations should thus be carefully avoided.

7.3.3. *Prototype Frameworks for Regime.* The market-oriented regime should protect innovations against commercial innovations only long enough to allow innovators to enjoy equal lead time to others who contribute a product of the same value to the market. The strategies put forward to do this include the automatic blockage of cloning. Or, more realistically, program behaviour and other industrial design elements should be protected against cloning long enough to avoid market failure. However, protection against cloning alone could be insufficient as largely similar innovations may not be deemed as clones. It may also be difficult without a registration system for second-comers to know the time length of anti-cloning protection. Additionally, no compensation would be rewarded by this system to any innovator whose development was a market failure, despite it offering substantial value to the market by providing an innovation which can be used by others to achieve commercial success.

The next step in the regime would be to follow automatic anti-cloning protection with an automatic royalty-bearing license. Firstly, clones would be blocked to give innovators time to develop a position in the market. Following this, mechanisms would be in place to ensure that compensation would be paid to developers who placed an innovation in the market which others sought to use, regardless of the initial product's commercial success. Again, practicality problems exist. It will still be difficult to assess when a blocking period ends and when the automatic license period begins without a registration system which describes the protected subject matter. Without a law encouraging or requiring standard terms in licenses, transactions costs for implementing licenses are likely to be high. Additionally, a reliable system to establish appropriate royalties for different type of use would be required.

Alternatively, in addition to the automatic anti-cloning protection, a registration approach could be developed. However, registration would be difficult to achieve because there is no intermediate design document which has been consistently prepared by software developers for either internal or external designs that could act as registration material. Reluctance by developers would be shown to register a design document disclosing all of their program's internal design elements which are currently able to be protected as trade secrets.

7.4. **Alternative Protection Strategies.**

7.4.1. *Registration and Automatic Licensing System for Innovations.* Anti-cloning legislation would be successful in solving the most serious protection requirements but it is still only a partial solution. To address a wider range of problems, a registration system for innovative sets of know-how included in software could be adopted. This would assist in creating a collection of advanced software which would help to further develop the industry in addition to possibly providing an exchange through which inexpensive transactions involving reusing software could occur. It would also provide a way to compensate innovators for disclosed innovations without having to spend the time and money of obtaining a patent. A challenge for this approach would be to ensure the framework would be adaptable as markets and technology evolve (Samuelson et al. 1994).

7.4.2. *Read Only Versions.* Often software manufacturers choose to sell a secondary version of their product with some functions removed at a lower price or even offer it

for free. Csorba (2002) examines what conditions are required to introduce a read-only version, with the writing function removed, and when it would be optimal to sell it for free. A good example is Adobe Acrobat which while being able to read and produce PDFs, is relatively expensive. Acrobat Reader, on the other hand, is only able to read PDFs but it is freely and legally downloadable from the Internet. This lower quality good will increase the market size which with the existence of network externalities, make the original full product more valuable, so consumers will have a higher willingness to pay. Publishers considering purchasing Adobe Acrobat to enable them to present their material in PDF form will look upon the investment more favourably if a large number of their prospective readers have access to Acrobat Reader. In effect, the higher price these consumers are willing to pay for the superior product subsidises the loss of providing the inferior product for free or very cheaply. In addition, if the amount of software sold increases, a Pareto-improvement will occur. Given sufficiently large network externalities versioning software may be a profitable option for the developer.

7.4.3. *Physical Protection Strategies.* Although it is uncommon for software developers to employ preventive control in their protection scheme, physical protection strategies do exist. These are laid out in detail in Collgerg and Thomborson (2002). In response to the threat of software tampering, *tamper-proofing* has become a possible means of protection. This prevents possible sizable financial losses caused by pirates extracting, modifying or otherwise tampering with encryption keys or other secret information required to be contained in e-commerce applications. Tamper-proofing will cause a program to malfunction when it detects that it has been modified. Other physical possibilities include *obfuscation* which attempts to transform a program into one which is more difficult to reverse engineer and *software watermarking* which places a copyright notice in the software code, allowing the software owners to assert their intellectual property rights. Similarly, *software fingerprinting* embeds a unique customer identification number into each copy to assist in tracking and prosecuting copyright violators. Despite these possibilities however, software developers accept that there is no protection scheme which will allow a product to survive a determined manual attack where human reverse engineers inspect the software for a long period of time. The most common form of technical protection form is simply a password activation scheme. However, this is easily overcome by a skilled hacker or anyone willing to search warez and other Internet sites for passwords.

8. Open Source Projects

8.1. **Nature of Programs.** A growing alternative source of software which by its very nature is not harmed by the increasing software piracy threat is Open Source or Copyleft software. These programs are public goods privately provided by volunteers. To copyleft a program, the programmer copyrights the program to himself and signs a General Public License which grants everyone the right to use, modify and distribute the program, but only if resulting modifications are granted similar rights. This can beneficially lead to substantial program development because any enhancements made do not have to be made by the original programmer (Mustonen 2003). Several OSS projects enjoy a reputation for innovation and reliability and have captured a significant market share from their commercial competitors (Bitzer and Schröder 2002). For example, Linux is estimated to have 21 million

users (Linux Counter 2006) and the Apache Web Server was used on 58 million sites, 63% of reachable web servers (Netcraft 2006). Advantages over traditional software provision include a greater ability for open source to access the pool of talented developers on the internet and the facility to use more private information. Some people, however, are reluctant to try out open source software, perceiving it to be less complete than closed source applications, which are believed to be easier to learn, contain more features and better documentation and are generally more user-friendly (Johnson 2002). Some disagree with its reliability and ability to innovate and claim that this software will always be restricted to niche areas (Hars and Ou 2002).

8.2. **Incentives for Programmers.** A computer programmer has the choice of being hired by a software firm, becoming an entrepreneur and starting a business or working in copyleft or open source programming (Mustonen 2003). Hars and Ou (2002) examine the weights of different motivations for programmers to partake in their third option. They find the motivations to be more complex than expected with both internal factors like intrinsic motivation, altruism and identification with a community and external factors such as direct compensation and anticipated return playing important roles. Hars and Ou also describe the different types of programmers in the open source community, with hobbyists and students generally exhibiting more internal motivations and salaried and contract programmers hoping to sell related products and services. Several authors examine the role of open source projects in signalling a programmer's abilities (Bitzer and Schröder 2002). This signal strengthens the provider's job prospects (Lerner and Tirole 2000 and Raymond 2000b) and improves reputations amongst the programmer community (Raymond 2000a and Diamond and Torvalds 2001).

8.3. **Effect on Monopoly.** Mustonen (2003) contributes to the literature by examining the impact of copyleft licensing on the development environment and on the market for programs. He extends traditional analysis by including a monopolist which supplies a copyright program. The existence of copyleft activity forces constraints on the monopolist in both the programmer labor market and in the consumer market with the substitute product. The copyleft program's role on the market depends largely on how expensive it is for a consumer to implement a program. If implementation costs are low, some consumers will choose the copyleft program, this must be taken into account by the monopolist when pricing. Copyleft can also create a barrier to entry for a prospective monopoly. For example, if there is a small market share and consumers have a low valuation of software, the monopolist may not enter and leave the entire market to copyleft. In the presence of a copyleft program, a monopolist is not able to exercise full monopoly power.

9. Conclusions

This paper has presented an overview of literature exploring several key issues in the computer software market. After a brief summary of characteristics of software, how copyright law relates to it and presenting some general information on software piracy, the existence and effects of network externalities was explored and it was found that with its presence software piracy could potentially have a positive effect on the market. The second important area considered in this paper was possible alternative protection strategies to copyright to allow software developers to recoup

their significant investment in product development and to provide incentives for future creations.

Copyright has traditionally been the form of protection used by software publishers to protect their work. Alternatively, requests for patents have been made over the years by developers, relying on the technological or idea characteristic of their programs. Difficulty in defining the nature of computer software as either a pure idea or as an expression has caused conflict between these two approaches of protection, and has lead to increased attention being placed on possible alternative regimes to protect the rights of developers.

However, the degree of optimal protection has been widely debated considering the balance of the negative and possible positive aspects of software piracy. While significant negative aspects can be clearly found, with the large financial costs to society presented in section four, many authors have presented persuasive arguments regarding the benefit of the larger network size caused by pirates, increasing the willingness to pay of legitimate purchasers and potentially overall profits for the publisher. Unanimous acceptance of this effect has not been reached and therefore the current global focus is generally on curbing software piracy.

Emerging from an industry greatly suffering from piracy is the interesting existence of open source projects. These non-commercial projects offer software for everyone to use and modify as they wish, so long as any resulting software remains open source. The increasing popularity of this software source, with its obvious immunity to the piracy threat, has presented commercial software with nontrivial competition.

The increasingly evident ineffectiveness of copyright laws in protecting the rights of computer software developers and the sizable costs to society of existing widespread piracy, with no consensus on its possible benefits, will mean that the analysis of motivations of piracy and possible alternatives for its prevention will continue to interest many researchers. Developing a regime to successfully lower the global piracy rate is important for adequately rewarding developers and ensuring the industry continues to provide useful innovations into the future.

References

Al-Jabri, I. and A. Abdul-Gader (1997), "Software Copyright Infringements: An Exploratory Study of the Effects of Individual And Peer Beliefs", *Omega*, 25(3); 335-44.

Anthes, G.H. (1993), "Study Cites Software Industry Growth, Piracy Problems", *Computerworld*, 27(13); 119.

Banerjee, D.S. (2003), "Software Piracy: A Strategic Analysis And Policy Instruments", *International Journal of Industrial Organization*, 21; 97–123.

Barret, D.R. (1993), "EC Policy On Intellectual Property And Standardisation – The Impact on the Computer And Telecommunications Industries", *Tolleys Computer Law and Practice*, 9(2); 46.

Bitzer, J. and P.J.H. Schröder (2002), "Bug-Fixing And Code-Writing: The Private Provision of Open Source Software", Discussion Papers of DIW Berlin, German Institute for Economic Research, Paper N^o 296.

Brynjolfsson, E. and C.F. Kemerer (1996), "Network Externalities in Microcomputer Software: An Econometric Analysis of the Spreadsheet Market", *Tims Institute of Management Sciences*, 42(12); 1627-47.

Business Software Alliance (n.d., a), *Software Piracy and the Law – Software Piracy in the United States*, Available on http://www.bsa.org/resources/upload/Software-Piracy-and-the-Law.pdf

Business Software Alliance (n.d., b), *Internet Software Piracy Fact Sheet*, Available on http://www.bsa.org/ireland/antipiracy/upload/Internet-Piracy-Factsheet.pdf

Business Software Alliance (2005), *Expanding the Frontiers of Our Digital Future: Reducing Software Piracy to Accelerate Global IT Benefits*, Available on http://www.bsa.org/idcstudy/pdfs/White_Paper.pdf

Chen, Y. and I.P.L. Png (1999), "Software Pricing and Copyright Enforcement: Private Profit vis-à-vis Social Welfare", *Proceedings of the International Conference on Information Systems*, 20; 119-23.

Cheng, H.K., R.R. Sims and H. Teegen (1997), "To Purchase or to Pirate Software: An Empirical Study", *Journal of Management Information Systems*, 13(4); 49-60.

Chiang, E. and D. Assane (2002), "Software Copyright Infringement Among College Students", *Applied Economics*, 34(2); 157-66.

Christensen, A.L. and M.M. Eining (1991), "Factors Influencing Software Piracy: Implications for Accountants", *Journal of Information Systems*, 5(1); 67-80.

Church, J. and N. Gandal (1992), "Network Effects, Software Provision, and Standardization", *Journal of Industrial Economics*, 60(1); 85-104.

Cohen, L.J. (1993), "Copyright, Multi-Media, Interoperability And Related Technologies – A Review", *Computer Law and Practice*, 9(1); 29.

Collgerg, C.S. and C. Thomborson (2002), "Watermarking, Tamper-Proofing, and Obfuscation – Tools for Software Protection", *IEEE Transactions on Software Engineering*, 28(8); 735-46.

Conner, K.R. and J.J. Rumelt (1991), "Software Piracy: An Analysis of Protection Strategies", *Management Science*, 27(2); 125-39.

Crampes, C. and J.J. Laffont (2002), "Copying And Software Pricing", IDEI Working Paper, Mimeo.

Csorba, G. (2002), "Read-only Versions for Free and for Profit: Functional Quality Differentiation Strategies of a Software Producing Monopoly", mimeo, Central European University, Budapest. Available online at http://www.serci.org/2002/Csorba.pdf.

David, P. (1985), "Clio And The Economics of QWERTY", *American Economic Review Papers and Proceedings*, 75; 1145-51.

Demsetz, H. (1970), "The Private Production of Public Goods", *Journal of Law and Economics*, 13; 293-306.

Gabella, G. and M.D. Picasso (1995), "PC Software Industry Lost $8.08 Billion To Pirates in 1994. Piracy Rate in the US Declines", *Information and Management*, 29(5); 285-8.

Gandal, N. (1994), "Hedonic Price Indexes For Spreadsheets And An Empirical Test For Network Externalities", *Rand Journal of Economics*, 25(1); 160-70.

Gandal, N. (1995), "Competing Compatibility Standards and Network Externalities in the PC Software Market", *Review of Economics and Statistics*, LXXVII; 599-608.

Givon, M., V. Mahajan, and E. Muller (1995), "Estimation of Lost Sales and the Impact on Software Diffusion", *Journal of Marketing*, 59(1); 29-37.

Gopal, R.D. and G.L. Sanders (1997), "Preventive and Deterrent Controls for Software Piracy", *Norsk Geologisk Tidsskrift*, 77(3); 29-48.

Gopal, R.D. and G.L. Sanders (2000), "Global Software Piracy: You Can't Get Blood Out Of A Turnip", *Communications of the ACM*, 43(9); 82-9.

Gwynne, P. (1992), "Stalking Asian Software Pirates (Jeffrey Seibach's Fight Against Software Piracy in Asia)", *Technology Review*, 95(2); 15-16.

Harkin, R. (1993), "A Design Right For Computer Programs?", *Tolleys Computer Law and Practice*, 9(3); 94-5.

Harrington, S. (1989), "Why People Copy Software and Create Computer Viruses: Individual Characteristics or Situational Factors?", *Information Resources Management Journal*, 2(3); 28-37.

Hars, A. and S. Ou (2002), "Working for Free? Motivations for Participating in Open-Source Projects", *International Journal of Electronic Commerce*, 6(3); 25-40.

Hebditch, D. (1986), "Pirate's Paradise (Shopping for Basic and Bootlegged Software in Hong Kong)", *Datamation*, 32(17); 30-1.

Hinduja, S. (2003), "Trends and Patterns Among Online Software Pirates", *Ethics and Information Technology*, 5(1); 49-61.

Im, J.H. and C. Koen (1990), "Software Piracy and Responsibilities of Educational Institutions", *Information and Management*, 18(4); 189-94.

Im, J.H. and P.D. Van Epps (1991), "Software Piracy and Software security in Business Schools: An Ethical Perspective", *ACM SIGMIS Database*, 22(3); 15-21.

Johnson, J. (2002), "Open Source Software: Private Provision of a Public Good", *Journal of Economics and Management Strategy*, 11(4); 637-62.

Katz, M.L. and C. Shapiro (1985), "Network Externalities, Competition, and Compatibility", *American Economic Review*, 75(3); 424-40.

Katz, M.L. and C. Shapiro (1999), "Antitrust in Software Markets", in J.A. Eisenach and T.M. Lenard (Eds.), *Competition, Innovation and the Microsoft Monopoly: Antitrust*, Dordrecht: Kluwer Academic Publishers; 29-40.

Kini, R.B., H.V. Ramakrishna and B.S. Vijayaraman (2004), "Shaping of Moral Intensity Regarding Software Piracy: A Comparison Between Thailand and U.S. Students", *Journal of Business Ethics*, 49(1); 91-104.

Koboldt, C. (1995), "Intellectual Property and Optimal Copyright Protection", *Journal of Cultural Economics*, 19(2); 131-55.

Landes, W. and R. Posner (1989), "An Economic Analysis of Copyright Law", *Journal of Legal Studies*, XVIII; 325-63.

Leibowitz, S.J. and S.E. Margolis (2005), "17 Famous Economists Weigh in on Copyright : The Role of Theory, Empirics and Network Effects", *Harvard Journal of Law and Technology*, 18(2); 435-57.

Lerner and Tirole (2000), "The Simple Economics of Open Source", *Journal of Industrial Economics*, 50(2); 197-234.

Linux Counter (2006). Online information retrieved 21 August 2006 from http://counter.li.org

Logsdon, J.M., J.K. Thompson and R.A. Reid (1994), "Software Piracy: Is It Related to Level of Moral Judgment?", *Journal of Business Ethics*, 13(11); 849-57.

Morgan, M.J. and D.J. Ruskell (1987), "Software Piracy – The Problems", *Industrial Management and Data Systems*, March-April; 8-12.

Mustonen, M. (2003), "Copyleft – The Economics of Linux And Other Open Source Software", *Information Economics and Policy*, 15(1); 99-121.

Netcraft (2002), Online information retrieved 21 August 2006 from www.netcraft.com

Peace, A.G., D.F. Galletta and J.Y.L. Thong (2003), "Software Piracy in the Workplace: A Model and Empirical Test", *Journal of Management Information Systems*, 20(10); 153-78.

Plant, A. (1934), "The Economic Aspects of Copyright in Books", *Economica*, 1; 167-95.

Poddar, S. (2002), "Economics of Software Piracy and its Global Impact", Retrieved May 8, 2006, from http://www.eco.rug.nl/SOM/SomSemC/Papers2002/16mei.pdf

Rahim, M.M., A.H. Seyal and M.N.A. Rahman (1999), "Software Piracy Among Computing Students: A Bruneian Scenario", *Computers and Education*, 32(4); 301-21.

Rasmussen, H.B. (2003), "Explaining Software Piracy", Student paper, CEBR.

Raymond, E.S. (2000, a), Homesteading the Noosphere", Revision 1.22, 2000/08/24, first version 1998. Available online at http://linux.cs.bbsbec.org/books/pdf/eric/homesteading.pdf.

Raymond, E.S. (2000, b), "The Cathedral and the Bazaar", Revision 1.51, 2000/08/24, first version 1997. Available online at http://csci.morris.umn.edu/UMMCSciWiki/pub/IS3221h/ReadingsFor26Feb2004/cathedral-bazaar.pdf

Richardson, G.B. (1997), "Economic Analysis, Public Policy and the Software Industry", working paper No. 97-4, Danish Research Institute for Industrial Dynamics.

Sacco, V.F. and E. Zureik (1990), "Correlates of Computer Misuse: Data From a Self-Reporting Sample", *Behaviour and Information Technology*, 9(5); 353-69.

Samuelson, P., R. David, M.D. Kapor, and J.H. Recihman (1994), "A Manifesto Concerning the Legal Protection of Computer Programs", *Intellectual Property Law Review*, 94; 327-450.

Shy, O. and J.F. Thisse (1999), "A Strategic Approach to Software Protection", *Journal of Economics and Management Strategy*, 8(2); 163-90.

Simpson, P.M., M. Banerjee and C.L. Simpson (1994), "Softlifting: A Model of Motivating Factors", *Journal of Business Ethics*, 13(6); 431-8.

Sims, R.R., H.K. Cheng and H. Teegen (1996), "Toward a Profile of Student Software Piraters", *Journal of Business Ethics*, 15(8); 839-849.

Slive, J. and D. Bernhardt (1998), "Pirated for Profit", *Canadian Journal of Economics*, 31(4); 886-99.

Software & Information Industry Association (2000), *SIIA's Report on Global Software Piracy 2000*, Available on http://www.siia.net/estore/GPR-00.pdf

Software & Information Industry Association (2004), *Software Use & The Law*, Available on http://www.siia.net/piracy/pubs/SoftwareUseLaw.pdf

Solomon, S.L. and J.A. O'Brien (1991), "The Effect of Demographic Factors on Attitudes toward Software Piracy", *Journal of Computer Information Systems*, 30(3); 168-81.

Swinyard, W.R., H. Rinne and A. Keng Kau (1990), "The Morality of Software Piracy: A Cross-Cultural Analysis", *Journal of Business Ethics*, 9(8); 655-64.

Takeyama, L.N. (1994), "The Welfare Implications of Unauthorized Reproduction of Intellectual Property in the Presence of Demand Network Externalities", *Journal of Industrial Economics*, 42(2); 155-66.

Torvalds, L. and D. Diamond (2001), *Just for Fun: The Story of an Accidental Revolutionary*, New York, HarperBusiness.

U.S. Government (2003), *Copyright Law of the United States of America and Related Laws Contained in Title 17 of the United States Code*, Circular 92.

Warren-Boulton, F.R., K.C. Baseman and G.A. Woroch (1994), "The Economics of Intellectual Property Protection for Software: The Proper Role for Copyright." Working Paper Industrial Organization Number 9411004, Economic Department of Washington University.

Wong, G., A. Kong and S. Ngai (1990), "A Study of Unauthorised Software Copying Among Post-Secondary Students in Hong Kong", *Australian Computer Journal*, 22(4); 114-22.

Wood, W. and R. Glass (1995),"Sex as a Determinant of Software Piracy", *Journal of Computer Information Systems*, 36(2); 37-40.

Amy Marshall, University of Canterbury, New Zealand.