Beyond Good and Evil: Why open source development for peer-to-peer networks does not necessarily lead to an Open Society, is as imbalanced as Copyright Law and definitely is not going to make you a better person

Prodromos Tsiavos
Department of Information Systems
London School of Economics and Political Science
Houghton Street, London WC2A 2AE, United Kingdom
P.Tsiavos@lse.ac.uk
Web page: http://is.lse.ac.uk

Ian Hosein
Department of Information Systems
London School of Economics and Political Science
Houghton Street, London WC2A 2AE, United Kingdom

Abstract
This paper interrogates the claims that open source development is an ideal form of regulatory development. We begin by presenting the literature that offers a framework of modalities of regulation where code, along with laws, markets, and norms shape and influence individual action. Within this framework, it is argued that for an Open Society we need Open Code. We present the processes through which the Gnutella protocol and the Limewire application are developed by deconstructing the mechanisms of participation and contribution of the individual developers. The techniques of monitoring, modularization and filtering that we identify appear to be inconsistent with open society promises. Instead we suggest a different framing, that of creating nests of interests, whose creators can find refuge from inhabitants of other nests. From that perspective, we suggest that we should stop referring to the war between Copyright and peer-to-peer networks as the battle between good and evil.

Keywords
Regulation, Open Source, Intellectual property

1. **Introduction: towards to Open Modalities**

The constitution of society is normally assumed to involve political human actors. As information systems researchers we endeavour to see the role of the technological. Yet normative questions arise, questioning the type of society we are building, and where decisions are being. The sources of control and influence over the individual, or modalities of regulation according to the Chicago School of the 1960s, include the market, laws, and social norms. Understanding the forces at play on individual autonomy, they argued, is important.

In the 1990s a New Chicago School approach was proposed by Lessig (1998). He added a further modality: Code, the constitution of technology. Code acts upon individuals, he warned, and the constitution of society and individual action could be affected by closed code. When this code is closed we may not see indirect forms of control: when other modalities shape code strategically, such as law (e.g. governmental policy on encryption) and market interests (e.g. Microsoft and Internet Explorer), it is difficult to trace accountability as the shaping is hidden, or closed off from open review.

The Open Society is a form of social organization that falls short of perfection but holds itself open to improvement. Epistemologically, actors' understandings are inherently imperfect; the and 'truth' is beyond our reach; but deliberation and discourse will help. It is characterized, ideally, by open access, however, and democratic principles. Whether it is the right under open-government laws to find out who has been shaping U.S. energy policies, or the right to participate in legislative initiatives through contacting your member of parliament to voice your concerns; this is the process through which legal regulation would ideally be formed. Over the past twenty years, the regulatory processes of many governments have been transforming to be 'open', transforming 'regulatory roadmaps' to improve clarity, and endeavouring for 'better regulation' through greater consultation and participation.

Open access to the constitution of the modalities may provide the difference between an open and closed society. To understand the role that technology plays as a modality, we need to understand its code. To ensure that the code does not conceal other interests and the indirect influence of other modalities, open code is required.

On the margin, open code reduces the reward from burying regulation in the hidden spaces of code. It functions as a kind of Freedom of Information Act for network regulation. As with ordinary law, open code requires that lawmaking be public, and thus that lawmaking be transparent. In a sense … open code is a foundation to an open society. (Lessig 2000, p.108)

With this 'open' code, we get a 'commons', a concept that Lessig (2001) approached while studying copyright-related issues. Lessig argues for a free commons where anyone can use a resource without permission, or that permission is granted neutrally (2001, p.12). In the case of Linux, he states

GNU/Linux is an 'open code' project. It is software that carries its source code with it, and requires that its source be kept available for others. The source code can be viewed and modified by a user; parts can be taken and used by other coders. It therefore builds a commons of (1) code, (2)knowledge, and (3) innovation upon that code. (Lessig 2001, p.54)

Openness and the creation of a commons will ensure that the Code modality works towards an Open Society.

This is democracy brought to code. An open code system can't get too far from the will of the users without creating an important incentive among some users to push the project a different way. ...
The users of an open code project are not therefore hostages. … They are not hostages to bad code -- the right to tinker is assured. And they are not hostages to strategic code -- open code can't behave strategically. (Lessig 2001, p.68)

Bearing in mind the other modalities, however, an Open Society will rely upon the market, law, and norms also conducting themselves openly.

This paper questions the assumption of open source leading to an open society. By looking at open source peer-to-peer (p2p) development we will show that open source code does not by its nature follow from open participation within a free commons. Next we look at copyright law and the rise of open source (§2). We will briefly discuss the methodology employed (§3), while the open source development discourse is presented in §4. We conclude in §5 that open source development is similar to the development of laws regarding copyright, which arguably do not follow the ideas of an open society either. Open source may be a good way of developing code, but we may not wish to idealise its regulatory role.

1. Copyright and Open Source

In the mid-1980s, with the emergence of personal computing, legal scholars were debating over the issue of how computer software should be legally protected (Reed and Angel 2002). The result is a number of updates to copyright law since (Lai 2002), arguably passed in the interests of copyright holders.

In this same period of time, technologists were having a different experience with this same problem. When software providers started blocking access to the source code that was previously available to everyone, a sense of discomfort grew amongst developers. A solution to this problem was to create a usage license to promote, rather than restrict, the availability and use of source code. This General Public License (GPL) allows open access to software’s source code while at the same time discouraging future developers from appropriating the code, and thus closing it. This was the dawn of a movement.

The Open Source movement believes that creativity comes out of sharing rather than restricting resources (Raymond 2001). Open source development was based on the concept of collaborative work of as many individuals as possible. The Internet provided an ideal “platform” for supporting such a mode of development; and a variety of software development tools were developed to support the collaboration of temporally as well as spatially dispersed developers.

The openness and decentralisation of the Internet are seen by the open source community as the primary means for supporting their cause. These same characteristics are often interpreted by law makers, software houses and jurists as rendering the Internet a significant threat to Intellectual Property Rights. Suddenly it was not just software piracy that was the problem: all kinds of content is disseminated over the Internet and since Napster emerged providing for file sharing, it was felt that Intellectual Property Law would never be the same again.

Some proclaimed the end of Copyright altogether (Barlow 1994); others called for amendments to the existing laws (Reed and Angel 2002); and others emphasised the importance of technology to solve the problem that technology has created (Johnson and

Post 1996). From the World Intellectual Property Organization (WIPO) treaties in 1995, to the U.S. Digital Millenium Copyright Act 1998, and the European Copyright Directive 2001, numerous approaches have emerged for dealing with copyright issues. New legislation combined with the Court decisions on Napster in the U.S. indicate that the Copyright holders have not only maintained their pre-Internet state of affairs; but have managed to increase their control (Lessig 2001). In the mid-90s the issue was how to protect Copyright Holders from the Internet (Vinje 1999). Today the issue is how to protect the Internet from the Copyright Holders (Lessig 2001); its open-ness is at stake.

The Gnutella protocol seems to be the quintessence of the post-Napster protection of the free Internet. The protocol supports a p2p application allowing the sharing of files amongst users employing a decentralised architecture. Some of the applications that use the protocol are also built in an open source fashion. In theory, all stakeholders or constituents should thus be satisfied, as it is a modality of regulation constituted by open society methods.

2. Methodology

One of our main objectives is to avoid oversimplifications and representations of technology as a monolithic entity. We sought to capture complexity rather than reducing it (Checkland 1981) through a detailed collection of the code development processes. As there is no set boundary in the phenomena under study a descriptive "case study" (Stake 1998) would be an oversimplified account. Instead, because the field under study remained in a state of constant negotiation and evolution, we see the process of development as a set of negotiations, or a socio-technological discourse that can be captured (Hosein 2002).

In order to identify the phenomena from which we would draw our data we "followed the actors" with a "rolling snowball" (Bijker 1995) letting them reveal the unit of analysis appropriate for our study. By tracing the development trajectory of the technological actors, i.e. the Gnutella/ LimeWire code-releases, as well as the structure of the fora, we were able to identify and follow the main involved parties.

3.1 Data Collection and Analysis Techniques

In a first stage we collected data related to the development of the Gnutella protocol. The sources included: web sites that were used for hosting forums and file repositories related to the development of the protocol that could be either archived or still operational; messages posted on discussion groups, forums and newsgroups; the design documents of the Gnutella protocol.

In a second stage we gathered material related to the Limewire application. The sources included: operational and archived web sites having been used for the development of the application; applications such as Concurrent Version Systems (CVS) or Bug reporting tools (such as Issuezila), design and implementation documentation and relevant press reports.

The data gathered covered a time span from early 2000 to late November 2002. The websites analysed were the archived sites of DSS Clip2, the Gnutella Developers (GnutellaDev), the Gnutella Next Generation (GnutellaNG), the Gnutella Developers

Forum (GDF), the LimeWire LLC (Limewire.com), the Limewire Open source (LimeWire.org) and the LimeGroup.

Other sources informed our research and also acted, at times of uncertainty, as forms of triangulation and verification (Lee 1991). These sources include websites such as Slashdot.org and WiredNews; IRC-mediated communications and private messages exchanged between the various developers. However we do not make extensive use of these sources in this representation; rather they all contributed to a hermeneutic process to increase our understanding (Boland 1985).

3. The Open Source Peer-to-Peer Discourse

In March 2000, Justin Frankel and Tom Pepper, employed by Nullsoft, a software firm, developed a small program, Gnutella. Using the underlying TCP/IP protocol stack, Gnutella creates a virtual infrastructure where each computer is the network. The p2p paradigm on which Gnutella is based results in the absence of any central points of control, i.e. "the client is the server is the network" (Oram 2001).

The original plan was for Gnutella to be released after reaching version 1.0 under a General Public License (GPL). Nullsoft was purchased by America On-Line (AOL) in June 1999 primarily for their other application, Winamp. Because of the legal problems that the Napster service was facing at the time, AOL declared Gnutella an "unauthorised freelance project" and decided to remove it from the Nullsoft website. At the time Gnutella was still in version 0.56; and the source code had not yet released publicly. This led to a reverse-engineering effort by some open source developers, and to further development.

The Gnutella communication protocol was then posted on a website, www.gnutella.nerdherd.net. At the time two developers created this environment that could sustain the project's initial phases.

The Clip2 Distributed Search Systems (DSS) was the first entity to start monitoring the operation and performance of the Gnutella network. Clip2 provided a forum for discussion between the Gnutella developers. Other forums that were used for the early development stages were the http://gnutellang.WEGO.com or the http://Gnutelladev.WEGO.com[1].

Problems began to arise within the existing developers' forums and the Clip2 DSS people decided to form another forum, known as the Gnutella Developers Forum (GDF). The reasons behind the formation of the GDF included the lack of an administrator in the WEGO groups; and that WEGO required a substantial amount of money for hosting the forums (GDF 2001). Another reason behind the migration of developers to the GDF was that other forums were hosted by particular Gnutella application vendors, and their neutrality was questioned (Clip2 2001). Although the GDF was initiated by Clip2 people, it was a conscious choice to select a platform other than that of the Clip2 web site for the hosting of the GDF. As the GDF founding document states:

> We believe placing the forum at a third-party provider gives it the best chance of neutrality and longevity. The "gdf_sysop@yahoo.com" eGroups account owns the GDF eGroup. Because of this, Clip2 has the ability to transfer administration rights to another party, such as a future developer organization. Another reason for using eGroups is the number of useful free tools provided by

[1] Both URLs are used for other purposes today.

> the service, including a chat system, file repository, link list, database system, poll service, etc. (Clip2 2001)

Clip2 ceased to exist in May 2001, but the GDF is still active with the latest additions to the Gnutella protocol having been posted in September 2002.

4.1 The Gnutella Protocol

Who participates, and under which role, in the development process is a first strong indicator of the openness of the development process.

The GnutellaNG was the working group that supported the development of the Gnutella protocol in its early stages. The participation of the developers to the evolution of the Gnutella protocol was administered in two phases. In the first stage, each developer had to apply for membership. An application was needed again for participation to those groups and the application format used was similar to that of GnutellaDev but the process of selection was rather different:

> The NG group works as follows:
>
> First we get all proposals and all ideas about the protocol, to see what everyone would want to see in the new protocol
>
> Then we establish together the functionnalities of the new protocol, and we create independant working groups each working on a different function of the protocol
>
> The independ working groups will be closed, and only the developpers part of each working groups will have access to discussion lists. As they're working, they will release recommendations for the gnutelladev working group, so as to provide a smooth upgrading path to the next generation protocol. (GnutellaNG 1999) [2]

The difference is located in the two steps filtering process of the participants to the working group. Everyone is invited to participate in the initial discussions but once formed, the individual working groups become closed discussion lists and the mobility of the developers is minimized. This is a layering or filtering pattern in the development of p2p projects that we have seen repeated in other cases of p2p projects (e.g. LimeWire) as well as closed source projects (e.g. Shareazaa). The differentiation in the levels of participation is really important if seen in conjunction with the lifecycle of the final development product: in the early stages of problem inception and abstraction, all discussion participants share a similar status, but when the problem deconstruction and the actual protocol design is to take place there is only a smaller number of skilled individuals that participates in the decision making process.

Other interesting elements in the way the whole process is organised relates to the thematic division of the working groups. The working groups are not pre-existing but created as a result of the discourse in the first stage of the development. The decisions about which functionalities are to be adopted are open and the end product is the result of synthesis of different opinions (“consensus”) rather than conflict (GnutellaNG 1999). The structure of the protocol, at least in the level of abstracting the main functionalities of the system, operated eventually as the blueprint for organising the working groups

[2] Original wording of the web site, with the spelling mistakes kept.

(GnutellaNG 1999). Thus we see the phenomenon, where the (assumed) structure of the final product feeds in the structure of the organisation that produces it.
The decision making process is said to be based on consensus. According to the Charter of the GnutellaNG:

> The keyword here is **consensus**. Developers joining this working group must understand the development of the next generation protocol means getting all people and all ideas together to provide the best possible protocol. So there won't be any "vote" system, nor will a specific idea be chosen over another just because the author has more "power". Everyone is equal. That means each functionality will be a merge of all ideas, getting the best part from each developer idea. (GnutellaNG 1999)

The lack of a voting or any other well defined system for supporting the decision making process led to a series of problems related with the efficiency of the working group and combined with some other issues, to the eventual disintegration of the group (Clip2 2001).

4.2 The LimeWire application

The documents of the GDF displaying the various proposals related to the evolution of the protocol shows the close interdependency between application and protocol development as well as the role that certain individuals have played in the development of the protocol. Out of the twelve proposals for changes, six come out from application developers, five from the same people that are the developers participating in the RFC-Gnutella project and only one from a non-affiliated person (GDF 2002). Moreover, the latest amendment to the Gnutella protocol, the so-called Gnutella2 (G2) is a suggestion by the Shareaza Group, a company that develops a proprietary closed-source application.
The involvement of the LimeWire people is substantial in the development of the protocol. This can be traced both in the involvement of the LimeWire senior developers in the finalisation of many of the Gnutella protocol documents and also in the participation of the LimeWire LLC team members in all kinds of Gnutella protocol and application forums. The revisions proposed with the introduction of the G2 protocol by the Shareaza people marks the appearance of another group that can influence the evolution of the protocol.

4.3 Modularization, Participation and Contribution Filtering for the LimeWire application

The LimeWire.org website constitutes the basis for the development of the LimeWire application. It comprises of six sections. The first one is titled “project info” and contains all the relevant documents for developing the LimeWire application.
The Development Resources section provides most of the co-ordination background the efforts of different developers is going to be based upon. The “Getting Started” section in particular incorporates a structure that funnels the effort of the individual developers towards the accomplishment of a single cause, i.e. the development of the LimeWire application (LimeWire.org 2002a). The steps that a developer takes in order to participate in the creation and improvement of LimeWire applications are as follows:
Download the LimeWire application in order to use the software and know its functionality

Download the source of the LimeWire application in order to have access to the source code
View all kinds of documentation available (source code, design documents, user manual)
Get involved by joining a mailing list, reporting a bug, or experiment with the source code. The issues related to the LimeWire source code are enlisted in the TODO section of the web site. (LimeWire.org 2002a)
A first reading of this list indicates that there are different levels of participation in the LimeWire project, a series of different projects to be taken care of and a list of priorities. All these different aspects of the overall LimeWire Project are enforced via a number of technical tools used to facilitate the participation process.
In terms of the levels of participation, everyone is invited to take part in the discussions in the various forums, since a mere registration suffices. The filtering in the participation process becomes more visible when the user is to shift from the level of simply participating in a conversation of a forum to that of getting involved in some kind of project. This is when the different “roles” a participant could undertake make their appearance (LimeWire.org 2002b). An individual could participate as an observer, a developer, a content developer or a project owner. Each one of these roles corresponds to different levels of rights and responsibilities.
Not all users can automatically assume any of the above roles. First, the user registers with LimeWire.org. Next she applies for a kind of membership. The user cannot apply directly for developer-status, but needs to be an ‘observer’ first, where she can only prove herself by participating in conversations, making comments or reporting bugs. The status of the user is defined by the owner of the project, which in terms of the LimeWire core or Graphical User Interface (GUI) is always one of the LimeWire LLC members. The more the role is closer to that of the observer the less the involvement of the user and its ability to influence the decision making process.
Even when a user becomes member of a LimeWire development project, he is not allowed automatically to participate in all projects; he needs to apply individually for each. This means that there is a modularization of the work required for the development of the final LimeWire application. Thus, there is a series of different projects that a user could participate in, the LimeWire file-sharing application being just one of them. The modularization of the tasks is to a great extent enforced by the structure of the software itself in the sense that the modules and the classes the users are asked to improve constitute the basis for the formation of different groups.
In each of these projects the user’s involvement is largely facilitated and directed by the tools used to perform certain tasks. For instance, in the case of reporting bug reports, there is the option of only providing reports for specific projects. Even in these cases the user is prompted to check the existing bug reports and only then submit a new one. The way the bug is to be reported is further standardised through the use of the Issuezilla, a web-based open source error report tool. All users are allowed to participate in the reporting of bugs. However, decisions concerning the future of these reports are taken only by those with a voting right. This happens through a Concurrent Versions System (CVS). CVS operates both as a co-ordination and as a collaboration system, structuring the way the code is viewed and altered as well allowing different developers to create different versions or modules of the system.

The decision making process is very much in the discretion of the LimeWire LLC team: "The LW core team decides, i.e., the full-time employees of Lime Wire LLC. This process is ad hoc and usually results in consensus."[3]

4. Conclusions

From what we have shown above, there are two sets of conclusions that may be drawn. First, indirect regulation may be a simplistic way of looking at the interaction between the modalities of regulation. Second, open source development processes are not necessarily open, nor necessarily lead to an Open Society.

5.1 On Indirect Regulation and *Regulatory Nesting*

In the work of many scholars that have dealt with the issue of regulation and new technologies (e.g. Biegel 2001; Lessig 2000; Murray and Scott 2002) we see the concept of indirect regulation as a commonly used mechanism for the final regulation of individual action. Law is most often seen as the primary, meta-regulator. Murray and Scott (2002) have advanced the model of indirect regulation even further, introducing hybrid regulatory forms, contesting the supremacy of law as a meta-regulator. Hosein, Tsiavos and Whitley (2002) have used the concept of regulatory ecologies to describe the way different modalities of regulation interact with each other. Our case study allows an even further deconstruction of the phenomenon: indirect regulation is indeed taking place and an ecology of regulations is created, but to conceptualise the whole phenomenon as the dominance of one modality of regulation over the others or as a single regulatory ecology constitutes an oversimplification.

In our discussion on copyright and open source we examined how a regulatory ecology is constructed to conclude that instead of that of the meta-regulator we could employ the metaphor of the regulatory nesting. Both Copyright Law and peer-to-peer technologies constitute regulatory modalities that serve different purposes and having different rationales. The objectives they pursue are mainly expressed through the Legal or Technological modality of regulation. But these are not the only modality used: Law may use of technological structures like Digital Rights Management systems, norms and markets, whereas LimeWire is based on a sui generis copyright license, the GPL, depends on user acceptance, and cultivates a series of norms regarding its development and use. In each case the law and technology respectively operate as the starting point, the nest around which other modalities of regulation playing a secondary but essential role are going to evolve.

5.2 On Open Source and the Open Society

The questioning of who is included and how they participate in the development process demonstrates that the constituents of the regulatory product are the ones that will be best accommodated in the regulatory nest. These tend to be developers in the case of the peer-to-peer open-source development, and the lawmakers and copyright holders in the case of Copyright Law.

[3] Informal communication (email) with Christopher Rhors, senior developer of LimeWire LLC

It is often argued that the powerful lobby interests of Hollywood prevent an open and just discourse regarding Copyright law (Ginsburg 2000); open source development processes are not by nature any better. The process is not open, the source code is not always a commons in that access is not given freely nor neutrally in accordance with Lessig (2001). Moreover, due to network externalities, just because source code is available does not mean that any individual may tinker with the code: participation within developers groups, all working on different decision-making forms (i.e. 'consensus',

Elaborating further on the issue of indirect regulation, the attempt of one modality to regulate another one -- especially when they belong to different nests -- can often cause the opposite result. The Napster case that was won in legal terms to be later lost in pragmatic terms if we take into consideration the amount of users currently sharing files illustrates exactly this point. It has been noted that p2p has exhibited a cockroach phenomenon (Hosein, Tsiavos, and Whitley 2002) since every legal action was coupled with a technological response. The analysis of the open source development process for peer-to-peer networks based on the regulatory nesting phenomenon clarifies the cockroach phenomenon: once a regulatory nest is assaulted its constituents whose needs cannot be accommodated by existing regulatory formation create new nests similar but more advanced than the original one. Even the internal process of the Gnutella development demonstrated this phenomenon: when the GnutellaNG and GnutellaDev started collapsing, the GDF emerged; when the GDF was not managing to meet its stated purpose of providing a new generation Gnutella protocol, the Shareaza people provided the Gnutella2 protocol.

To evaluate open source development for peer-to-peer networks in ethical terms is at least naïve. The question of balancing of rights becomes equally absurd: The Gnutella/LimeWire development structures accommodate the needs of mainly the developers and subsequently the users in the same way that Copyright law satisfies the needs of the disseminators that have better access to the lobbying process for the law formation. Both regulatory nests use rhetoric that the others are ignoring artists' rights but the latter are under or misrepresented in both cases. Where the 'user' is represented in either nest also remains a question for further research.

With regard to the openness of the open source development this very much remains in the realms of ideological papers, technical papers that are fascinated with the results of the development process and economics papers that discuss the incentives behind open source development or the implications of the "gift economy" (Feller and Fitzgerald 2002). Perhaps it is time to move beyond these accounts and attempt to understand why open source development for peer-to-peer networks does not necessarily equal to an open society, is as imbalanced as Copyright Law and definitely is not going to make you a better person.

5. References

Barlow, J.P. "The Economy of Ideas". From Wired, Issue 2.03, March 1994.

Biegel, S. Beyond our control? : confronting the limits of our legal system in the age of cyberspace. Cambridge, Mass.: MIT Press, 2001.

Bijker, W. E. Of bicycles, bakelites, and bulbs : toward a theory of sociotechnical change Inside technology. Cambridge, Mass: MIT Press, 1995.

Boland, R. J. "Phenomenology: A Preferred Approach to Research on Information Systems." In Research Methods in Information Systems, ed. E. Mumford et. al., 193-201. North-Holland: Elsevier Science Publishers, 1985.

Checkland, P. Systems Thinking, Systems Practice. Chichester: John Wiley, 1981.

Clip2 DSS. FAQ on founding of Gnutella Developer Forum, Clip2 DSS, 2002.

Feller, J. and Fitzgerald, B. (2002) Understanding Open Source Software Development, London, Addison-Wesley.

GDF. The Gnutella Proposals, available at http://groups.yahoo.com/group/the_gdf/files/Proposals/proposals.htm, 2002.

Ginsburg J (2000)Copyright Use and Excuse on the Internet, 24 Colum.-Villanova Law Journal on Law and the Arts.

GnutellaNG, Welcome to gnutellaNG, 1999, available at http://web.archive.org/web/20000816001207/http://gnutellang.wego.com/.

Hosein, I. "A Research Note on Capturing Technology: Towards Moments of Interest.", ed. To be published by the International Federation for Information Processing 8.2: Kluwer, 2002.

Hosein, I., Tsiavos, P., and Whitley, E. "Regulating Architecture and Architectures of Regulation: Contributions from Information Systems." A paper delivered at the 17th British and Irish Law, Education and Technology Association, Free University, Amsterdam, 2002.

Johnson D & Post D, Law and Borders—the Rise of Law in Cyberspace, 48 Stan. L. Rev.

Lai, S. Copyright Protection of Computer Software in the UK, Hart Publishing, 2002.

Lee, A. S. "Integrating Positivist and Interpretive Approaches to Organizational Research." Organization Science, Volume 2, Number 4, 1991, pp. 342-365.

Lessig, L. "The New Chicago School." Journal of Legal Studies, Volume 27, Number June, 1998, pp. 661-691.

Lessig, L. Code : and other laws of cyberspace. New York, N.Y.: Basic Books, 2000.

Lessig, L. The future of ideas : the fate of the commons in a connected world. 1st ed ed. New York: Random House, 2001.

Limewire.org (2002a). Getting Started,LimeWire LLC. 2002.

LimeWire.org (2002b). Project Help for Developers, LimeWire LLC. 2002.

Murray, A., and Scott, C. Working Paper -- Controlling the New Media: Hybrid Responses to New Forms of Power. London: LSE and Research School of Social Science, Australian National University2002.

Oram, A. (ed.) (2001) Peer-to-peer: Harnessing the Benefits of a Disruptive Technology, Cambridge O' Reilly

Raymond, E. (2001). The Cathedral and the Bazaar: Musings on Linuxand Open Source by an Accidental Revolutionary, O'Reilly: Sebastopol, CA.
Reed, C. J. Angel: *Computer Law*, 4th ed. Oxford: Oxford Unviersity Press, 2002.
Stake, R. E. "Case Studies." In Strategies of Qualitative Inquiry, ed. Norman K. Denzin and Yvonna S. Lincoln, 86-108. London: Sage Publications, 1998.

Vinje, T (1999) Copyright Imperilled? European Intellectual Property Review, issue 4, pp 192-207.