Dieser Artikel ist Teil des

# Open Source Jahrbuchs 2008


erhältlich unter www.opensourcejahrbuch.de.

Die komplette Ausgabe enthält viele weitere interessante Artikel. Sie können diesen und andere Artikel im Open-Source-Jahrbuch-Portal kommentieren oder bewerten: www.opensourcejahrbuch.de/portal/. Lob und Kritik sowie weitere Anregungen können Sie uns auch per E-Mail mitteilen.

# Monopolelemente bei freier Software

MATTHIAS BÄRWOLFF*

*(CC-Lizenz siehe Seite 281)*

Die Quelltexte freier Software sind zwar unter der GPL-Lizenz uneingeschränkt verfügbar, andere Kontexte sind jedoch weit mehr ausschließbar gegenüber Dritten. Die Kontrolle über ein Freie-Software-Projekt und dessen spezifische Ressourcen führt dazu, dass auch freie Software Marktmacht ausüben kann. Dies ist nicht moralisch verwerflich, sondern im Gegenteil unabdingbare Voraussetzung für die Möglichkeit, Profite auf Seiten der Produzenten zu erwirtschaften und unabhängig von Subventionen zu sein. Kategorien von Software, bei der keine Möglichkeiten bestehen, Profite mit freier Software zu erwirtschaften, werden demnach traditionell überwiegend von proprietären Anbietern dominiert, da diesen über Lizenzgebühren ein Teil des erwirtschafteten Mehrwerts zufällt.

*Schlüsselwörter:* Freie Software · Monopol · Monopolistischer Wettbewerb

## 1 Einleitung

Freie Software wird häufig als der Antipol schlechthin zu proprietärer Software verstanden. So bemerkt etwa Moglen (2007):

> „Die Qualität von Software hat von 1980 bis 1990 rapide abgenommen, [. . .] [denn] Monopole produzieren bekanntermaßen minderwertige Güter zu hohen Kosten und verhindern Innovationen. [. . .] [Aber] eine kleine und unorganisierte Community von Leuten, die Software zur *gemeinsamen* Nutzung entwickelten, hat die Begrifflichkeiten der Debatte neu definiert, vortreffliche Güter zu Kosten von Null produziert und mit der nunmehr unaufhaltsamen Auflösung des Monopols begonnen."

* Ich danke Wolfram Riedel für seine scharfsinnigen und außerordentlich hilfreichen Anmerkungen zu einer früheren Version dieses Artikels.

Freie Software und „das Monopol" sind demnach unüberbrückbare Gegensätze – zwei Pole eines Kontinuums, mithin ohne jedwede Schnittmenge. Die Prämissen dieser Aussagen sind in etwa die folgenden: Freie Software unterliegt per Definition der *GNU General Public License* (GPL) oder einer anderen Open-Source-Lizenz, die die fortwährende Verfügbarkeit des Quelltextes von Software sicherstellt.[1] Mithin ist also jeder Versuch, aus dem Verkauf von solcherart lizenzierter Software Kapital zu schlagen, zum Scheitern verurteilt. Schließlich erlaubt es die Lizenz jedermann, die Software weiterzugeben, wenn auch zu denselben Lizenzbedingungen. Der Marktpreis wäre also ob der praktisch unbegrenzten Verfügbarkeit der Software gleich den marginalen Kosten des Vertriebs. Was die Nutzer der Software angeht, so könnten diese die Software nach Belieben verändern oder Dritte die Software verändern lassen. „Lock–ins", also das Ausgeliefertsein an einen bestimmten Anbieter oder eine bestimmte Technologie, sind daher ebenso abwegig wie das Erzielen von Monopolerlösen aus dem Verkauf von Software.

Aber schon Lessig (2002) hat bemerkt, dass die Produktion von freier Software insoweit vergleichbar ist mit anderen Produktionsprozessen, als dass bei ihr private Ressourcen mit öffentlich verfügbaren Allmenderessourcen zur Schaffung von ökonomischem Mehrwert vermischt werden. Dabei landen zumindest Teile des Mehrwerts bei den Produzenten. Wir möchten in diesem Artikel noch einen Schritt weiter gehen und fragen: Sind freie Software und Monopolmacht tatsächlich unvereinbar? Sind Freie-Software-Projekte vielleicht letztlich gar nicht so verschieden von proprietärer Software, als dass sie ebenso Marktmacht entwickeln und ausüben können?

Unsere Intuition dabei ist ebenso einfach wie die oben beschriebene und hier in Frage gestellte. Eine Open-Source-Lizenz führt zwar zur unbeschränkten Verfügbarkeit des *Codes*, nicht aber der von spezifischen *Kontexten*. Hierzu zählen schon allein der Name einer Software, dessen markenrechtlicher Schutz ganz unabhängig von der Nutzungslizenz ist, aber auch die Beherrschung von komplementären Fertigkeiten oder die Existenz bestimmter komplementärer Artefakte auf Seiten von Produzenten und Anwendern. Daraus folgt, dass es sowohl einen Ort der Kontrolle über ein jedes Freie-Software-Projekt gibt als auch eine daraus resultierende Marktmacht sowie das Potenzial, mittels Komplementäreffekten Monopolrenten zu erwirtschaften.

## 2 Freie Software im Kontext von monopolistischem Wettbewerb

In diesem Abschnitt schweifen wir kurz ab und betrachten die wechselseitige Abhängigkeit von Wettbewerb und Monopolen, um das Konzept des monopolistischen Wettbewerbs später auf freie Software anwenden zu können. Die häufig gezeichnete Dichotomie zwischen den Begriffen Wettbewerb und Monopol ist nämlich – genauso

1 Wir beschränken uns im Rahmen dieses Artikels auf die Betrachtung GPL-lizenzierter Software und verzichten auf eine Ausdehnung der Betrachtung auf BSD-artige Lizenzen oder andere Lizenzen, die von der *Open Source Initiative* anerkannt sind (http://www.opensource.org/licenses).

wie deren populäre ethische Bewertung – nicht nur fragwürdig, sondern häufig auch irreführend.

Monopole sind in einem ganz elementaren Sinne entscheidend für das Funktionieren einer dezentralen Wirtschaft. Wettbewerb und Monopole bedingen einander, wie Chamberlin (1950) treffend ausführt:

> „Ein wesentlicher Teil freien Unternehmertums ist das Bemühen eines jeden Geschäftsmannes, [nicht homogene Güter in vollkommenem Wettbewerb zu produzieren, sondern] sein eigenes Monopol aufzubauen, es möglichst weit auszudehnen und es zu verteidigen gegen die Bemühungen anderer, ihre Monopole auszubauen. Es gibt keine Tendenz, derzufolge diese Monopole durch den Wettbewerb zwischen Unternehmen verschwinden, ganz im Gegenteil: Sie sind genauso ein Teil des Gesamtbilds wie der Wettbewerb, der sie beschränkt.
>
> Die Tatsache, dass Produkte differenziert sind, bringt die Probleme der Vielfalt und Auswahl zum Vorschein und macht deutlich, dass *vollkommener Wettbewerb in keinster Weise als ‚ideal' anzusehen ist für volkswirtschaftliche Wohlfahrtsbetrachtungen*. In vielen Fällen wäre es praktisch unmöglich, solchen Wettbewerb herzustellen, selbst wenn er denn wünschenswert wäre. Geschäfte, zum Beispiel, könnten sich nicht alle an einem Ort befinden, genauso wie Unterschiede zwischen Schauspielern, Sängern, Freiberuflern und Geschäftsleuten sich nicht vermeiden lassen. Und selbst dort wo es möglich wäre, wäre es nicht erstrebenswert, Produkte übermäßig zu standardisieren. Verschiedene Geschmäcker, Vorlieben, Einkommen, Wohnorte und Nutzungen von Gütern bezeugen schließlich die Notwendigkeit von Produktvielfalt. Das ‚Ideal des vollkommenen Wettbewerbs' muss also ersetzt werden durch eines, das beides beinhaltet, Monopol und Wettbewerb, und dies unbeschadet der Frage, wie viel und welche Art von Monopol, mit welcher Art von sozialer Kontrolle.
>
> Weiterhin ist es unerlässlich, die ‚ideale' Abstimmung der Kosten für Herstellung, Werbung und Vertrieb, wie die konventionelle Analyse von Preis und Produktionsmenge auch, explizit als Teil des Wohlfahrtsoptimums zu begreifen. Diese zusätzlichen Elemente sind fraglos weitaus eher veränderlich als die Preise selbst, und doch ist es die Standardannahme der ökonomischen Wohlfahrtsmaximierung, dass erstens Produkte gegeben sind und es zweitens keine Vertriebs- und Werbekosten gibt, [was natürlich falsch ist]." (S. 213 ff., Fußnoten weggelassen, Betonung im Original, Übersetzung des Autors)

Monopole sind also nicht etwas, das es „natürlicher Weise" in einer Wirtschaft nicht gibt, sondern im Gegenteil gerade eine Folge von Wettbewerb unter verschiedenen Anbietern.

Wettbewerb wird nur in den Lehrbüchern vereinfacht gleichgesetzt mit festgelegten Gütern, deren Preise sich dann im Wettbewerb den marginalen Kosten nähern. In der Realität gibt es vielmehr eine Reihe weiterer veränderlicher Variablen, über die sich Wettbewerb definiert, unter anderem eben die Produkte selbst.[2] Weiterhin werden in idealisierten Betrachtungen Kosten für Marketing, Vertrieb und Werbung entweder als unnütz und schädlich abgetan oder gleich gänzlich vernachlässigt. Damit unterschlägt man jedoch eine in der Realität immens wichtige Angelegenheit. Über strategische Entscheidungen und Aufwendungen in diesen Bereichen entscheiden sich häufig Erfolg oder Misserfolg eines Unternehmens, nicht über die Preisgestaltung entlang hypothetischer Nachfragekurven für festgesetzte Produkte.

Dass dies im Bereich der freien Software ökonomisch betrachtet anders sein soll, ist zwar eine intuitiv verlockende Vorstellung, empirisch jedoch völlig unhaltbar.

Heute ist weitgehend unstrittig, dass freie Software, genau wie jedes andere Gut auch, entweder durch Monopolrenten oder durch Subventionen finanziert wird. Die Möglichkeit für Produzenten von freier Software, aus dem untrennbaren Kontext einer Software Profite zu erzielen, also aus Komplementärgütern oder -dienstleistungen, wie spezieller Anpassung der Software oder Unterstützung bei deren Einrichtung und Betrieb, wurde in der Literatur sehr ausführlich beschrieben (siehe etwa Grand et al. 2004; Bärwolff 2006) und braucht hier nicht weiter erörtert zu werden.[3]

Freie Software, deren Entwicklung und Vertrieb nicht durch solche Monopolrenten finanziert wird, ist häufig finanziert durch Steuersubventionen – also öffentliche Aufwendungen im universitären oder anderweitig steuerfinanzierten Forschungsbereich – oder durch Firmen, deren kommerzielle Software im Markt gescheitert ist und die daraufhin ihre Software als freie Software veröffentlicht haben. Eine wichtige Rolle spielen zudem Entwickler, deren Opportunitätskosten hinreichend gering sind (Lancashire 2001) und die durch ihre Mitarbeit an freier Software in einer Art „zeitlich umgekehrter Quersubventionierung" soziales Kapital akkumulieren und potenziellen Arbeitgebern ihre Fähigkeiten signalisieren (Lerner und Tirole 2002).[4]

Im folgenden Abschnitt möchten wir diesen empirischen Befunden folgen und zeigen, dass freie Software in vielerlei Hinsicht sehr ähnlich zu proprietärer Software ist. Schon die Produktion freier Software ist häufig weit weniger frei, als der Name es erscheinen lässt, was dazu führt, dass auch in den entsprechenden Produktmärkten Monopolmacht ausgeübt werden kann.

2 Bei Kosten von Null (wie im Falle von GPL-lizenzierter Software) wären ja sonst *alle* Variablen per Definition gegeben – eine völlig absurde Vorstellung.

3 Wir möchten nur kurz anmerken, dass es in diesem Bereich interessante Entwicklungen gibt, etwa die viel diskutierte TiVo-Festplatten-Set-Top-Box, bei der die GPL-lizenzierte Software des Geräts derart mit der Hardware verknüpft ist, dass diese keine modifizierten Versionen der Software akzeptiert. Die Freiheiten der GPL sind hier, zumindest im Bezug auf die TiVo-Hardware, nur theoretisch gegeben, praktisch aber irrelevant, da Änderungen der Software zwar möglich sind, nicht aber auf der Zielplattform lauffähig.

4 Siehe zur Bedeutung von geringen Opportunitätskosten für informelle Produktionsprozesse auch Pinker (2006).

## 3 Freie-Software-Projekte als Firmen

Um unserem Argument einen tragfähigen Ausgangspunkt zu geben, ist es zunächst sinnvoll, festzuhalten, dass es auch bei freier Software einen institutionellen und organisatorischen Kontext gibt, in dem Kontrolle über die Entwicklung und den Vertrieb einer Software ausgeübt werden kann und oft auch wird. Insofern unterscheidet sich ein Freie-Software-Projekt nicht großartig von einem kommerziell motivierten proprietären Softwarehersteller. Beides sind Firmen im Sinne von Coase (1937), als dass es einen mehr oder minder definierten Raum gibt, in dem die Allokation von Ressourcen nicht über Preise, sondern durch Befehlsketten in Hierarchien organisiert wird. In kommerziellen Unternehmen sind es die Aktionäre, mittelbar vertreten durch die Führungsebene, die die angestellten Mitarbeiter anleiten und kontrollieren. Bei freier Software ist es zumeist der ursprüngliche Entwickler, der zum einen die Namensrechte an der Software hält und zum anderen die Kontrolle und Führung über das Projekt ausübt.

Wenn die freie Software einer kommerziellen Firma „gehört“,[5] dann gilt prinzipiell selbes wie im Kontext einer jeden anderen Firma: Bezahlte Angestellte entwickeln die Software auf Weisung ihres Arbeitgebers. Nur selten gibt es dann überhaupt externe Parteien, die Rechte irgendwelcher Art an dem resultierenden Produkt haben. Externe Beiträge zur Software werden entweder nicht berücksichtigt oder nur aufgenommen, wenn die Entwickler ihre Rechte daran der Firma überschreiben.[6] Und auch, wenn die freie Software von einer verteilten Community außerhalb eines formalen Firmenkontexts entwickelt wird, gibt es praktisch immer eine Hierarchie, in der die oberen Ebenen Kontrolle über die unteren ausüben, zumindest aber darüber, welche Arbeiten ihren Weg in die freie Software des Projekts finden (Priddat 2006). Es ist mithin nicht unüblich, dass Beiträge von Entwicklern abgelehnt werden, und dies nicht etwa, weil sie anderen konkurrierenden Beiträgen qualitativ nachstehen, sondern weil es gerade bei großen Projekten überhaupt nicht mehr ohne Weiteres möglich ist, die Qualität von Beiträgen objektiv und nach rein technischen Kriterien zu beurteilen. Status wird dann häufig wichtiger als Leistung, ganz entgegen der gängigen Behauptung, in Freie-Software-Projekten würden Beiträge *alleine* nach objektiven technischen Qualitätskriterien beurteilt.[7] In Freie-Software-Projekten entstehen also die selben Ineffizienzen wie in jeder anderen Firma auch, sie entkommen nicht der

5 Siehe etwa das MySQL-Datenbankmanagementsystem der schwedischen Firma *MySQL AB* oder den Java-Application-Server *JBoss* der mittlerweile zu *Red Hat, Inc.* gehört.

6 Für *MySQL AB* ist es zum Beispiel nur dann möglich, externe Beiträge mit in die MySQL-Software aufzunehmen, wenn die externen Entwickler ihre Rechte daran an MySQL abtreten. Das MySQL–Geschäftsmodell, die Software sowohl unter der GPL als auch unter proprietären Lizenzbedingungen zu vermarkten, bedingt es, dass alle Rechte an der Software bei *MySQL AB* liegen.

7 O'Mahoney und Ferraro (2004) betrachten das Debian-Projekt und dessen Probleme, mit einer wachsenden Mitgliederzahl umzugehen. An einem Punkt wurde das Projekt gar völlig geschlossen für neue Mitglieder, als klar wurde, dass die Organisation des Projekts nicht einmal mehr operativ, geschweige denn effizient mit der wachsenden Mitarbeiterzahl umgehen konnte.

Logik der Coaseschen Transaktionskosten, nur weil sie sich anderen Idealen als der Profitmaximierung verpflichtet fühlen.[8]

Ein häufig geführtes Argument besagt nun, ein jeder könne ob der frei vorhandenen Quelltexte einem Freie-Software-Projekt Konkurrenz machen, etwa falls dieses sich zu sehr kommerziellen Interessen beugt oder in anderer Weise „unfreier" wird, aber auch falls jemand das Projekt einfach nur in eine Richtung weiterentwickeln möchte, die das Projekt ablehnt. Durch diese prinzipiell vorhandene Möglichkeit des *forking*[9] löst sich jedoch nicht das eben beschriebene Problem. Zum einen sind Entwickler für freie Software nicht unbegrenzt verfügbar,[10] und zum anderen gelten auch für das neue Projekt dieselben strukturellen Einschränkungen der Freiheit wie für das alte. Auch hier wird jemand die Kontrolle über das Projekt ausüben, auch hier wird es eine Entscheidungshierarchie geben, und auch hier werden entsprechende Ineffizienzen auftreten.

Dazu kommt, dass die geistigen Eigentumsrechte an der Software, zumindest bis zum Punkt der Abspaltung des neuen Projekts, bei den ursprünglichen Entwicklern verbleiben, auch wenn die Software unter einer Open-Source-Lizenz jedem frei zur Verfügung steht. Das bedeutet mithin, dass derjenige, der das neue Projekt ins Leben ruft, nur eben jene Rechte an der freien Software erwirbt, die ihm qua Open-Source–Lizenz zustehen.[11] Weiterhin wird die neue Software sowohl aus praktischen Gründen als auch aus markenrechtlichen Gründen unter einem neuen Namen firmieren müssen, dem ursprünglichen Projekt also nicht unter demselben Namen Konkurrenz machen können. So besteht also die prinzipielle Möglichkeit, einem Freie-Software-Projekt Konkurrenz mit dessen eigenen Quelltexten zu machen, es besteht aber auch ein sehr wirkungsvoller und begrenzter Ort der Kontrolle über ein Projekt, der sich nur schwer untergraben lässt. Wenn es nun also bei freier Software de facto einen „Eigentümer" gibt, der oft sogar eine juristische Person ist, dann gibt es damit auch jemanden, der Monopolmacht ausüben kann. Die Frage ist nun, ob und in welcher Weise solche Monopolmacht ausgeübt wird.[12]

8 Mitunter wird argumentiert, die Verteilung von Aufgaben in Freie-Software-Projekten passiere mehr oder minder selbstorganisiert, indem offene Aufgaben als solche gekennzeichnet werden und dann von Entwicklern mit freien Kapazitäten übernommen werden, ohne dass es hierfür einer zentralen Koordinierung bedürfe (Heylighen 2007). Dies ändert jedoch nichts an den institutionalisierten hierarchischen Kontrollprozessen in Freie-Software-Projekten, über die sich entscheidet, welche Beiträge in das Projekt aufgenommen werden.

9 Mit *forking* bezeichnet man das Abspalten eines neuen Software-Projekts aus einem gegebenen Projekt.

10 Anders ausgedrückt, die Ressourcen eines Projekts und die Fähigkeiten der Mitarbeiter sind ob ihrer Spezifizität nicht beliebig kopierbar (Wernerfeld 1984).

11 Unter den Lizenzbedingungen der GPL etwa kann man, da die neue Software ein abgeleitetes Werk der alten bleibt, nur die Erweiterungen, nicht aber das gesamte Werk unter einer anderen Lizenz weitervertreiben. Dies ist beispielsweise die Basis des Geschäftsmodells der oben schon erwähnten Firma *MySQL AB*. Falls jemand die MySQL-Software erweitern und unter einer anderen Lizenz als der GPL weitergeben möchte, so kann er hierfür die Software statt unter der GPL auch unter einer proprietären Lizenz von *MySQL AB* erwerben, da diese ja sämtliche Rechte an der Software hält.

12 Die Tatsache, dass freie Software praktisch immer zu Kosten von Null abgegeben wird, tut dieser

## 4 Zwei kurze Fallbeispiele für Monopolelemente bei freier Software

Sobald ein Freie-Software-Projekt eine gewisse Marktrelevanz entwickelt, beginnt es den obigen Ausführungen zufolge, über potenzielle Marktmacht zu verfügen.[13] Zwei Fallbeispiele mögen dies veranschaulichen:

*Firefox vs. Iceweasel* Der Webbrowser *Firefox* der *Mozilla Foundation* ist fast schon ein Paradebeispiel für die Monopolmacht von Produzenten freier Software. Trotz Kosten von Null für die gegebene freie Software werden hier Profite erzielt und Konkurrenten diskriminiert: Profite werden erzielt durch die Partnerschaft mit *Google*, die jährlich etwa 50 Millionen US-Dollar an Einnahmen bringt.[14] Und Konkurrenten, die auf Basis der Quelltexte von *Firefox* unter der GPL–Lizenz das Produkt ändern und weitervertreiben möchten, sind an strenge markenrechtliche Richtlinien der *Mozilla Foundation* gebunden, denen zufolge niemand die Quellen des Browsers ändern und dann unter dem Namen *Firefox* weitergeben darf. Das Debian-Projekt etwa, eine der ältesten und bedeutendsten Linux-Distributionen weltweit, ist dadurch in Konflikt mit der *Mozilla Foundation* geraten. Als man im Jahre 2006 kleine Verbesserungen und Anpassungen an *Firefox* vornehmen wollte, deren Aufnahme in das Hauptprojekt von den Mozilla-Entwicklern abgelehnt wurde, verbot die *Mozilla Foundation* es dem Debian-Projekt, die resultierende Software weiterhin unter dem Namen *Firefox* in ihre Linux-Distribution aufzunehmen. Seither nennt sich diese Software bei *Debian Iceweasel*, eine für Anwender zunächst gewöhnungsbedürftige Änderung. Nun sind *Iceweasel* und *Firefox* zwar noch zum ganz überwiegenden Teil dieselbe Software, es sind aber eben doch unterschiedliche Produkte mit unterschiedlichen Namen, unterschiedlichen Vertriebswegen und mithin unterschiedlichen Anwendergruppen.

Dieses Beispiel entspricht ziemlich genau dem Modell des monopolistischen Wettbewerbs von Chamberlin (1950). Die *Mozilla Foundation* möchte verhindern, dass andere Anbieter ihr Monopol auf den Browser *Firefox* untergraben,

---

Frage keinen Abbruch. Schon Chamberlin (1950) bemerkte sehr treffend, dass sich Monopole nicht unbedingt alleine über Preise, schon gar nicht über Profite definieren:

> „[P]rofite sind lediglich ein Element in [Monopol-]Situationen; Preise, Diskriminierungspraktiken, Dienstleistungen in allen ihren Aspekten, Investitionen usw. können stark beeinflusst sein durch Monopolelemente, obgleich keine exzessiven Profite erzielt werden." (S. 195 f., Übersetzung des Autors)

13 Wir möchten bemerken, dass wir in keiner Weise die Relevanz von freier Software in unkommerziellen, oft akademischen Kontexten als „Spielwiese" für Innovationen (Bresnahan 1998) in Abrede stellen.

14 Siehe etwa http://weblogs.mozillazine.org/mitchell/archives/2007/01/the_mozilla_foundation_achievi.html [6. Feb. 2008].

andere hingegen möchten den Bedürfnissen ihrer speziellen Zielgruppen gerecht werden und machen dem Projekt *Firefox* Konkurrenz – entweder, indem sie wie bei *Iceweasel* die Quelltexte des Browsers als Basis für ein abgeleitetes, aber verschiedenes Konkurrenzprojekt verwenden oder indem sie ein ganz eigenes Konkurrenzprodukt zu *Firefox* entwickeln.[15]

*RHEL vs. CentOS* Dass verschiedene Produkte auch bei großer Ähnlichkeit über Monopolmacht im oben genannten Sinne verfügen, wird am Beispiel der beiden Linux-Distributionen *Red Hat Enterprise Linux* (RHEL) und *CentOS* deutlich. Ersteres ist ein Produkt des kommerziellen Unternehmens *Red Hat, Inc.*, letzteres eine praktisch identische Version desselben Produkts, erzeugt aus den Quelltexten von RHEL mit dem Ziel, vollständige Kompatibilität zu RHEL zu erreichen. Dennoch haben beide Projekte sehr unterschiedliche Zielgruppen und Anwender, da der Kontext beider Projekte sehr verschieden ist. *Red Hat* verkauft Abonnements, Zertifizierungen und kommerziellen Support, während die CentOS-Community lediglich informelle Hilfe über Mailinglisten, Foren, Chat etc. anbietet. Obwohl also beide Projekte praktisch dieselbe Software vertreiben, sind sie für die Anwender nur sehr unvollständige Substitute füreinander, eben ob des sehr unterschiedlichen Kontexts.

Ähnliches gilt für die seit 2006 bestehende Initiative von *Oracle Corp.*, bei der die unter der GPL stehenden Quelltexte von RHEL unter dem Namen *Unbreakable Linux* zusammen mit entsprechenden Support-Verträgen vertrieben werden. Aber auch hier gilt, dass *Oracle* andere Kunden anspricht als *Red Hat*, schon alleine, weil *Red Hat* deutlich mehr Expertise bezüglich seiner Software hat als *Oracle*, die die Software lediglich unter einem anderen Namen weitervertreiben. *Oracle* spricht denn auch überwiegend Kunden an, die schon Kunden der Datenbank-Software von *Oracle* sind und denen vor allem an einem reibungslosen Zusammenspiel ihrer Datenbank und dem darunter liegenden Linux-System gelegen ist.

Neben den genannten auf RHEL basierenden Distributionen gibt es noch eine Unmenge weiterer Linux-Distributionen, alle mit unterschiedlichen Fokussen, Eigenheiten und Zielgruppen.[16] Hier wird deutlich, dass gerade weil der Preis als Variable im Wettbewerb von konkurrierenden Anbietern freier Software irrelevant, da Null ist, der Wettbewerb sich auf andere Bereiche, speziell die konkreten Eigenschaften der Software konzentriert. Auch hier zeigt sich wieder eine deutliche Analogie zu den Ausführungen von Chamberlin (1950): Wettbewerb findet nur selten über Preise statt, sondern vielmehr über das Produkt selbst, auch und gerade bei freier Software.

15 Siehe etwa den Konqueror-Browser des KDE-Desktop-Projekts unter http://www.konqueror.org [6. Feb. 2008].

16 Siehe etwa die unter http://distrowatch.com gelisteten Linux-Distributionen.

Die genannten Beispiele machen deutlich, dass freie Software *nicht* gleichbedeutend ist mit vollkommenem Wettbewerb, sondern eher dem oben beschriebenen Modell von monopolistischem Wettbewerb entpricht. Jede freie Software hat ihre über die eigentlichen Quelltexte hinausgehenden Eigenheiten und jedes Projekt Eigenschaften, die sich nur schwer nachbilden lassen. Damit hält jedes ein Monopol für ein mehr oder weniger eng gefasstes Marktsegment und steht als solches im Wettbewerb mit anderen Projekten, die ihrerseits Monopole halten. Jedes hat eine eigene strategische Ausrichtung, und jedes ist in einen unterschiedlichen Kontext eingebettet, sowohl auf Seiten der Produktion als auch auf Seiten der Anwendung.

## 5 Monopolelemente freier und proprietärer Software im Vergleich

Freie und proprietäre Software unterscheiden sich voneinander, insbesondere bezüglich der Stärke der Monopolmacht, die sie ausüben können. Freie-Software-Monopole sind gemeinhin deutlich schwächer als proprietäre (Gruber und Henkel 2006). Während es bei freier Software praktisch nur die Rechte am Namen der Software und die Kontrolle über eine Entwickler-Community sind, die ursächlich zum Schutz des Monopols genutzt werden können, kommt bei proprietärer Software noch die Kontrolle über die Quelltexte qua restriktiv ausgeübtem Urheberrecht und Geheimhaltung dazu, häufig in Verbindung mit Patenten für bestimmte Funktionalitäten der Software (Samuelson und Scotchmer 2002). Bei proprietärer Software ist es also für Konkurrenten wesentlich schwerer, identische oder direkt abgeleitete Produkte zu erstellen, ja überhaupt nur die Funktionalitäten oder bestimmte Standards einer Software nachzuahmen. Stattdessen gibt es aber auch einen höheren Anreiz, sinnvolle Alternativen zu bestehenden Monopolen zu entwickeln, diese genießen ja dann eben solchen Monopolschutz.[17]

Bei freier Software ist eine derartige Dominanz eines einzelnen Unternehmens nur schwer vorstellbar. Dafür ist es aber eben auch kaum möglich, mit freier Software direkt jene Monopolgewinne zu erwirtschaften, die für risikobehaftete Innovationen und professionelles Marketing gemeinhin notwendig sind. Gewinne können hier, trotz Monopol, oft nur über die oben schon angesprochenen Komplementäreffekte wie Anpassungen oder spezielle Erweiterungen der Software erfolgen, was aber nicht

17 Es gibt Fälle, in denen die Monopole bei proprietärer Software aufgrund von Netzwerk- und Skaleneffekten derart stark sind, dass es für Wettbewerber praktisch unmöglich ist, dem Monopol direkt Konkurrenz zu machen (Bresnahan 1998). Dann besteht die einzige sinnvolle Möglichkeit, das bestehende Monopol anzugreifen, darin, die Relevanz des Bereichs, in dem sich der Monopolist bewegt, selbst zu untergraben. Wenn also etwa webbasierte Software à la *Google* die heute noch dominante Desktop-Software à la *Microsoft Office* ablösen würde, dann käme es zu einer solchen von Bresnahan (1998) beschriebenen epochalen Verschiebung. *Microsoft* argumentiert derzeit tatsächlich, dass genau diese epochale Verschiebung nunmehr stattfindet (Rule et al. 2007).

immer praktikabel oder überhaupt möglich ist.[18] Diese Schwierigkeiten treffen mithin häufig genau auf jene Software zu, die am besten auf die Bedürfnisse von einfachen Benutzern zugeschnitten *wäre* – dort bräuchte es ja keine Anpassungen oder anderweitige Dienstleistungen mehr. Hier bliebe also nur die Finanzierung qua Subventionen, was angesichts fehlender Marktsignale allerdings nur ein bedingt taugliches Mittel der Ressourcensteuerung bleibt.[19]

## 6 Fazit

Freie Software weist durchaus Monopolelemente auf. Sie steht zwar per Definition unter einer Lizenz, die die völlig ungehinderte Weitergabe der Software mitsamt ihrer Quelltexte sichert, dennoch ist ein Freie-Software-Projekt ob der vorhandenen zentralisierten Kontrollmechanismen eine Firma im Sinne von Coase (1937): Es gibt ein Machtzentrum über die begrenzten Ressourcen für die Entwicklung der Software des Projekts sowie Namen und relevante Kontexte, woraus eine Marktmacht für das Software-Produkt selbst folgt.

Freie Software ist also nicht frei in einem unbedingten und absoluten Sinne, sie ist genauso an Namen, Personen und Kontexte gebunden wie proprietäre Software auch.

Die Monopolmacht eines Freie-Software-Projekts ist allerdings deutlich eingeschränkter als die eines proprietären Herstellers. Dies ist zwar aus einer statischen Perspektive günstig für die Anwender, nicht jedoch, wenn man die dynamischen Vorteile von Monopolen und der Aussicht auf solche berücksichtigt. Der oft gepriesene Vorteil freier Software – die Beschränkung von Monopolrenten wegen der Unmöglichkeit, Standards von nur einem Anbieter zu kontrollieren – ist denn gleichzeitig auch ihr größter Nachteil: Der Mehrwert landet zum überwiegenden Teil beim Konsumenten, kaum etwas verbleibt beim Produzenten. Nur wenn es seitens der Produzenten gelingt, aus den vorhandenen Kontexten Profit zu schlagen, lohnt es sich für diese, freie Software zu produzieren. Sonst bleibt freie Software auf Subventionen und Steuerfinanzierung angewiesen, was die Übermittlung von Präferenzen der Anwender hin zu den Produzenten und damit den sinnvollen und zielgerichteten Einsatz von Ressourcen ganz enorm erschwert.[20]

18 Wie schon in Fußnote 12 erwähnt: Ein Monopol muss nicht zwingend mit exzessiven Profiten einhergehen.

19 Es gibt populäre Paradigmen, denen zufolge alle unsere Erwägungen müßig sind, weil Menschen in vielen Fällen freiwillig und in einer neuartigen „sozialen" Art und Weise Werte schaffen, selbst und vor allem solche, die in Markt- oder Firmenkontexten schlicht nicht umsetzbar wären (von Hippel 2005*a*; Benkler 2006). Die ebenfalls populäre Steigerung dieses Modells verzichtet gar auf jegliche positive Anteile und konstruiert einen rein normativen „ethischen Imperativ", demzufolge es gilt, anderen durch proprietäre Software nicht „zu schaden". Wir halten solche Modelle, die empirisch nur dünn belegt und in ihren Implikationen letztlich eher utopisch als politisch sind, jedoch nicht auf die von uns betrachteten Softwaremärkte anwendbar.

20 Wir möchten am Rande anmerken, dass steuerfinanzierte und subventionierte Entwicklung von freier Software sich dort anbietet, wo die konkreten Funktionalitäten einer Software hinreichend spezifi-

Freie Software ist also nicht die institutionelle Lösung für sämtliche Gebiete der Softwareentwicklung, sondern wird auch in Zukunft überwiegend Bereiche bedienen, in denen Anpassungen und Serviceleistungen bezüglich einer Software strukturbedingt notwendig sind, zuvörderst bei Software mit Plattform-Charakter. Denn nur dort, wo bei freier Software aus Monopolmacht Profite generiert werden können, kann es auch zu Innovationen und Investitionen in Software-Produkte kommen. Dort wo dies nicht gelingt, bleiben die ökonomischen Anreize, gute Software-Produkte als freie Software zu bauen, unvollständig.

## Literatur

Benkler, Y. (2006), *The Wealth of Networks: How Social Production Transforms Markets and Freedom*, Yale University Press. http://www.benkler.org/Benkler_Wealth_Of_Networks.pdf.

Bresnahan, T. (1998), New Modes of Competition and the Future Structure of the Computer Industry, *in* 'Competition, Convergence, and the Microsoft Monopoly', Bd. Progress and Freedom Foundation Volume, Kluwer. http://www.stanford.edu/~tbres/research/pff.pdf.

Bärwolff, M. (2006), 'Tight Prior Open Source Equilibrium', *First Monday* **11**(1). http://www.firstmonday.org/issues/issue11_1/barwolff/index.html.

Chamberlin, E. H. (1950), *The theory of monopolistic competition: A re-orientation of the theory of value*, 6. Aufl., Harvard University Press. Erstmals 1933 veröffentlicht.

Coase, R. H. (1937), 'The nature of the firm', *Economica* **4**, S. 386–405.

Grand, S., von Krogh, G., Leonard, D. und Swap, W. (2004), 'Resource Allocation Beyond Firm Boundaries: A Multi-Level Model for Open Source Innovation', *Longe Range Planning* **37**, S. 591–610.

Gruber, M. und Henkel, J. (2006), 'New ventures based on open innovation—An empirical analysis of start-up firms in embedded Linux', *International Journal of Technology Management* **33**(4), S. 356–372.

Heylighen, F. (2007), Warum ist Open-Access-Entwicklung so erfolgreich? Stigmergische Organisation und die Ökonomie der Information, *in* B. Lutterbeck, M. Bärwolff und R. A. Gehring (Hrsg.), 'Open Source Jahrbuch 2007. Zwischen freier Softwareentwicklung und Gesellschaftsmodell', Lehmanns Media, Berlin, S. 165–180. http://www.opensourcejahrbuch.de.

Lancashire, D. (2001), 'Code, Culture and Cash: The Fading Altruism of Open Source Development', *First Monday* **6**(12). http://firstmonday.org/issues/issue6_12/lancashire/index.html.

ziert sind. Dies war etwa bei *Linux* der Fall, hier ging es primär um eine freie Implementierung des POSIX-Standards. Ebenso sind die GNU-Tools überwiegend Implementierungen von vorher bestehender Software. Völlig neue Produktinnovationen sind in diesem Bereich aber eben nicht zu erwarten.

Lerner, J. und Tirole, J. (2002), 'Some Simple Economics of Open Source', *Journal of Industrial Economics* **50**(2), S. 197–234. http://www.people.hbs.edu/jlerner/simple.pdf (Arbeitspapier aus dem Jahre 2000).

Lessig, L. (2002), Open Source Baselines: Compared to What?, *in* R. Hahn (Hrsg.), 'Government Policy toward Open Source Software', AEI-Brookings Joint Center for Regulatory Studies, Washington, DC, Kapitel 4, S. 50–68, notes at 99–105. http://www.aei.brookings.org/publications/abstract.php?pid=296.

Moglen, E. (2007), 'The Global Software Industry in Transformation: After GPLv3', http://www.archive.org/details/EbenMoglenLectureEdinburghJune2007StreamingVideo384kbits, transcript at http://www.archive.org/details/EbenMoglenLectureEdinburghJune2007text. Lecture in Edinburgh, Scotland, 26th June 2007.

O'Mahoney, S. und Ferraro, F. (2004), Managing the boundary of an 'open' project, IESE Research Papers D/537, IESE Business School. http://www.iese.edu/research/pdfs/DI-0537-E.pdf.

Pinker, R. (2006), 'From Gift Relationships to Quasi-markets: An Odyssey along the Policy Paths of Altruism and Egoism', *Social Policy and Administration* **40**(1), S. 10–25.

Priddat, B. P. (2006), Open Source als Produktion von Transformationsgütern, *in* B. Lutterbeck, M. Bärwolff und R. A. Gehring (Hrsg.), 'Open Source Jahrbuch 2006. Zwischen Softwareentwicklung und Gesellschaftsmodell', Lehmanns Media, Berlin, S. 109–121. http://www.opensourcejahrbuch.de.

Rule, C. F., Kanther, J. S., Smith, B. L., Snapp, M., Heiner, Jr., D. A., Holley, S. L. und Pepperman II, R. C. (2007), Memorandum of Points and Authorities of Microsoft Corporation in Opposition to Certain Plaintiff States' Motions to Extend the Final Judgments, Memorandum submitted to the US District Court of Columbia in civil action no. 98-1233 against Microsoft, Microsoft Corporation. http://www.microsoft.com/presspass/download/legal/settlementproceedings/11-07MemoranduminOpposition.pdf.

Samuelson, P. und Scotchmer, S. (2002), 'The Law and Economics of Reverse Engineering', *Yale Law Journal* **111**, S. 1575–1663. http://www.yalelawjournal.org/pdf/111-7/SamuelsonFINAL.pdf.

Wernerfeld, B. (1984), 'A resource-based view of the firm', *Strategic Management Journal* **5**(2), S. 171–80.

von Hippel, E. (2005*a*), *Democratizing Innovation*, MIT Press, Cambridge, MA. http://web.mit.edu/evhippel/www/democ.htm.