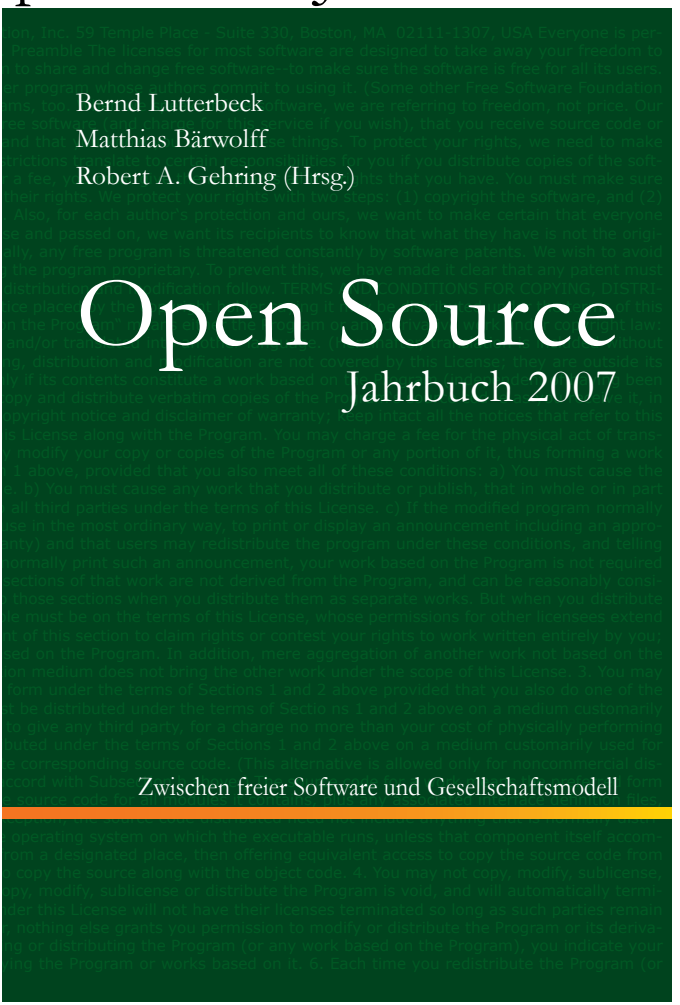
Dieser Artikel ist Teil des

Open Source Jahrbuchs 2007



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.

Die GPL v3 – ein Schutzschild gegen das Damoklesschwert der Softwarepatente?

LINA BÖCKER

Große Reformen hinterlassen Spuren. So auch die fast vollständige Neufassung der *General Public License (GPL)*.[1] Insbesondere patentrechtliche Fragen, eines der meist umstrittensten Gebiete im Bereich der Rechtsinformatik, wurden erstmals ausdrücklich geregelt und in den Lizenztext in Form der neuen Ziff. 11 aufgenommen. Ziel dieser Änderungen war es, bekannten Risiken, die aus dem Aufeinandertreffen von freier Software und Patentrecht entstehen, vorzubeugen und mehr Rechtssicherheit zu schaffen. Ziel dieses Beitrags ist es, diese Gefahren aufzuzeigen und zu überprüfen, ob die neuen Regelungen tatsächlich Abhilfe schaffen oder ob sie nur ein Placebo sind. Letzten Endes ist aber festzuhalten, dass auch durch die Neufassung der GPL nur die Symptome gelindert werden können. Wirklich helfen kann nur ein neu überdachtes, wenigstens europaweit harmonisiertes Patentrecht.

Schlüsselwörter: GPL · Lizenzrecht · Patentrecht · Softwarepatente

1 Bleibt alles anders – die neue GPL

Verständlicher und juristisch einfacher sollte sie werden, die neue GPL. „Die größte Gemeinschaftsarbeit, die jemals an einer Softwarelizenz vorgenommen wurde" (Jaeger 2006), hat zu einer fast vollständigen Neufassung der Bedingungen geführt, die das Ergebnis angeregter Diskussion aller Beteiligten ist. Besonders kontrovers wurden Kopier- (*Digital Rights Management*, *DRM*) und Patentschutz für Computerprogramme debattiert. Kritik an den auf Patente bezogenen Klauseln kam vor allem aus den Reihen der Softwareindustrie und hat, was Softwarepatente angeht, zu erhebli-

1 Dieser Artikel berücksichtigt den Diskussionsstand zur GPL v3 bis zum 27. Februar 2007.

chen Änderungen im Vergleich zur ersten Entwurfsfassung vom Januar 2006 geführt.[2] Während sich die GPL v2 in ihrer Präambel und auch noch der erste Diskussionsentwurf der GPL v3 vehement gegen Softwarepatente in jeglicher Form ausgesprochen haben, kritisiert die neue Fassung nur noch auf Universalrechner bezogene Patente.[3]

Der erste Diskussionsentwurf enthielt, anders als die „alte" GPL v2, die explizite Erteilung eines einfachen Nutzungsrechts am Patent an jeden Erwerber der Software, also eine Art automatische Patentlizenz (FSFE 2006). Diese Konstruktion wurde durch den verpflichtenden Verzicht auf die Geltendmachung von Patentansprüchen ersetzt. Einen übermäßigen Eingriff in die Rechte des Patentinhabers, die schließlich Eigentumsrechte sind, wollten die Verfasser verhindern.

Das Thema „Patente für computerimplementierte Erfindungen" ist in Industrie und Wissenschaft schon fast ein „Klassiker", wird doch seit mehr als 20 Jahren darüber diskutiert, ob und unter welchen Voraussetzungen Software bzw. Computerprogramme patentierbar sein sollen. Wer genau hinschaut, stellt fest, dass es einen ähnlichen Streit schon gab: Nach der Gründung des Zweiten Deutschen Reichs 1871 diskutierte man darüber, das Patentwesen insgesamt abzuschaffen, da es kein einheitliches gab und eine Vereinheitlichung schwierig erschien. In diesem Kontext kam auch die Diskussion auf, wie mit der Patentierbarkeit für Stoffzusammensetzungen auf dem Sektor der chemischen Industrie umgegangen werden sollte (Nack 2004, S. 774). Hier wurde mit Zustimmung der Industrievertreter ein Patentierungsverbot beschlossen, das trotz nachträglich auftretender Schwierigkeiten 90 Jahre bestehen blieb.

2 Die Diskussion über die Softwarepatente in der EU

Die Diskussion über das Ob und Wie eines Patentschutzes für Computerprogramme ist in den vergangenen zwei Jahren von einer rechtsdogmatischen zu einer rechtspolitischen geworden.[4] Für Außenstehende ist die Debatte insbesondere um den Entwurf einer europäischen *Richtlinie über die Patentierbarkeit computerimplementierter Erfindungen*[5] immer schwerer zu verstehen, was unter anderem damit zusammenhängt, dass viele der Streitigkeiten lediglich auf terminologische Ungenauigkeiten zurückzuführen sind. Zwar wurde der Richtlinienentwurf im Juli 2005 gekippt, das Thema ist jedoch weiterhin auf dem Tisch. Dass noch immer eifrig debattiert wird, liegt daran, dass der Zustand ohne Richtlinie ebenso wenig den Vorstellungen der „Kriegsparteien" entspricht wie der Zustand mit, da ohne ein Gesamtkonzept die Patentierungsvoraussetzungen insgesamt unklar und uneinheitlich sind. Das birgt natürlich auch die Gefahr einer Ausweitung des Patentschutzes über das notwendige Maß hinaus.

2 Der Begriff „Softwarepatente" wird im Rahmen dieses Artikels synonym für die „Patentierbarkeit computerimplementierter Erfindungen" verwendet.

3 Siehe dazu in der GPL-Präambel, letzter Absatz oder bei FSFE (2006).

4 Ausführlich dazu bei Horns (2000, Abs. 1–80).

5 Erster Entwurf im Amtsblatt der EU Nr. C 151 E vom 25. Juni 2002, S. 129, eine gründliche Diskussion der kontroversen Entwürfe findet sich bei Metzger (2003).

Die Frage nach der Möglichkeit der Patentierbarkeit von Computerprogrammen stellt sich in der oft behaupteten Form eigentlich gar nicht.[6] Denn bei der Frage nach der grundsätzlichen Patentierungsmöglichkeit kann es nicht um das äußere Erscheinungsbild einer Erfindung gehen (Nack 2004).[7] Zu prüfen ist vielmehr, welche erfinderische Leistung hinter dem in Frage stehenden Produkt steckt. Das gilt selbstverständlich auch für Computerprogramme. Grenzt das Patentgesetz in § 1 Abs. 3 Nr. 3 Computerprogramme als solche vom Schutz aus, so ist nichts anderes als das gemeint. Allein die Tatsache, dass es sich bei der Erfindung um ein Computerprogramm handelt, spricht weder für noch gegen ihre Patentierbarkeit (Nack 2004). Die erfinderische Leistung, die Grundlage des Patents ist, muss auf einem Gebiet der Softwaretechnologie liegen, dass als „Technik" anerkannt ist (Nack 2004, S. 772). Gemeint sind Steuerungs- und Regelungstechniken, die den Ablauf technischer Prozesse beeinflussen, wie etwa die Elektronik in einem Auto.[8]

3 Patentrechtsverletzungen und Open-Source-Software

Die Abgrenzungsschwierigkeiten zwischen patentierbaren Programmen und jenen, die lediglich Computerprogramme als solche sind, führen zu unkalkulierbaren Risiken, was zwischen Patentgegnern und -befürwortern weitgehend unumstritten ist. Diese Risiken, die im Folgenden dargestellt werden, können sich bei Open-Source-Software um ein Vielfaches heftiger auswirken als bei proprietärer Software, da bei Letzterer der nicht zugängliche Quellcode gründliche Patentrecherchen verhindert.[9]

3.1 Grundsätzliche Voraussetzungen einer Patentverletzung

Gefahren für Entwickler und Anwender entstehen durch das Risiko einer Patentverletzung beim Umgang mit freier Software, die Schadensersatz- und Unterlassungsansprüche nach sich ziehen kann.[10]

Verletzungstatbestände

Handelt es sich bei dem in Frage stehenden Softwarepatent um ein *Verfahrenspatent* im Sinne von § 9 Nr. 2 PatG, ist also ein bestimmter technischer Ablauf geschützt, so genügt jegliche Anwendung des patentierten Verfahrens, um eine Patentverletzung zu begehen. Angewendet wird ein Verfahren dann, wenn es bestimmungsgemäß gebraucht wird, d. h. die wesentlichen Schritte, die zum verfahrensgemäßen Erfolg

6 Siehe dazu Nitschke (2002). Zur grundsätzlichen Patentfähigkeit so genannter softwarebezogener Erfindungen siehe Metzger (2000). Zum grundsätzlichen Streit um den Schutz computerbezogener Erfindungen eingehend bei Weyand und Haase (2004).

7 Zur Eingrenzung des Begriffs *Softwarepatente* siehe Horns (2000).

8 Zum entscheidenden Merkmal der Technizität s. Horns (2000, Abs. 2 ff.) und Klopmeier (2002, S. 65ff.).

9 Siehe auch Horns (2000) sowie Weyand und Haase (2004).

10 Rechtsprechungsbeispiele bei Pierson (2004).

führen, müssen verwirklicht werden. Dies ist nur dann nicht verboten, wenn der Anwender berechtigter Benutzer des Verfahrens ist oder sich die Wirkung des Patents nicht auf seine Handlungen erstreckt. Bei Software reicht also das Ausführen auf einem Computer, um eine Patentverletzung herbeizuführen (Busse 2003, § 9 RdNr. 83 ff.).

Lange Zeit beschränkte sich der Softwareschutz auf solche Verfahrenspatente, da Computerprogramme in der Regel reale Prozesse nachbilden. Die jüngere Rechtsprechung lässt jedoch auch einen *Erzeugnisschutz* zu. Der Patentanspruch richtet sich dann nach dem digitalen Medium, auf dem das Programm gespeichert ist. Dies ist insoweit patentierbar, als es das Mittel zur Ausführung des technischen Verfahrens ist. Bereits das Herstellen, Anbieten oder Inverkehrbringen des geschützten Erzeugnisses ist hier rechtsverletzend,[11] also auch ein Anbieten über das Internet. Ebenfalls zur Benutzung eines Erzeugnispatents gehören die Einfuhr und der Besitz.

Nicht nur die direkte Verwendung des patentierten Erzeugnisses oder Verfahrens ist eine Patentverletzung. Eine mittelbare Benutzung im Sinne des § 10 PatG liegt bereits vor, wenn Mittel angeboten oder geliefert werden, die die unberechtigte Benutzung der Erfindung ermöglichen (Busse 2003, § 10 RdNr. 13).

Ausnahmetatbestände

Jeder dieser beschriebenen „Tatbestände" stellt eine Patentverletzung dar und führt zu Schadensersatz- bzw. Unterlassungsansprüchen. Nicht jedoch dann, wenn einer der patentrechtlichen Ausnahmetatbestände erfüllt ist, die sich aus § 11 PatG ergeben. Im Umgang mit Open-Source-Software kommt hier vor allem ein Handeln zu Versuchszwecken im Sinne von Nr. 2 in Frage. Das rechtfertigt die Verwendung patentierter Erzeugnisse dann, wenn es sich um die Verwendung von Code zur Neuentwicklung eines anderen Produkts handelt. Die Verwendung des Patents zu Versuchszwecken ist allerdings nur dann freigestellt, wenn es um die Gewinnung von mehr Erkenntnissen über die patentierte Erfindung geht.[12] Das ist dann nicht der Fall, wenn die Erfindung, also der patentierte Code, zum Mittel der Forschung gemacht wird. Die Weiterentwicklungsmöglichkeit darf Gegenstand dieser Patentverwendung sein, so dass die Open-Source-Weiterentwicklung grundsätzlich unter diesen Tatbestand fallen kann, auch wenn sie proprietären Code als Basis nimmt. Zu beachten ist dann aber, dass auch das Ergebnis der Forschung keine Patentverletzung darstellen darf.

Rechtsfolgen

Patentrechtsverletzungen haben Unterlassungsansprüche und im Verschuldensfalle – hier dem zivilrechtlichen Verschulden, welches auch die Fahrlässigkeit umfasst – auch Schadensersatzansprüche zur Folge. Es besteht einerseits die Gefahr einer Hemmung

11 Beispielsweise BGH GRUR 1959, 125 – Textilgarn; GRUR 1956, 77 – Rödeldraht; BGH GRUR 1979, 149, 151 – Schießbolzen; GRUR 1996, 109 – klinische Versuche.

12 Näher dazu Busse (2003, § 11 RdNr. 17 f.) mwN.; sowie GRUR 1996, 439, 446 und GRUR 1996, 120.

der Innovationsgeschwindigkeit bei der Geltendmachung von Unterlassungsansprüchen, andererseits birgt das Risiko einer Patentverletzung eine erhebliche finanzielle Bedrohung insbesondere für kleinere Entwickler, die auf eigene Faust arbeiten.

3.2 Mögliche Risiken der Patentierbarkeit von Software für Open Source

Was aber macht gerade den Patentschutz so gefährlich für die Open-Source-Entwicklung, dass er eines der Hauptthemen in der Diskussion um die GPL-Reform darstellt?[13] Zunächst bestehen gegen den Patentschutz von Software einige grundsätzliche Bedenken, unabhängig davon, ob es sich um Open Source oder um proprietäre Software handelt. Ein erteiltes Patent ist ein staatlich garantiertes Monopol, das es ermöglicht, eine Technologie unter Ausschluss jedes Dritten zu nutzen (Hufnagel 2002). Dieses Monopol besteht für 20 Jahre nach der Anmeldung der Erfindung beim Patentamt. Betrachtet man demgegenüber das Tempo, mit dem Entwicklungen in der Computerindustrie voranschreiten, erscheint diese Schutzdauer unverhältnismäßig lang. Schließlich sind ganze Programme oftmals bereits nach einem halben Jahr veraltet, ganz zu schweigen von einzelnen Programmteilen oder Updates und Patches. Die Konsequenz ist, dass die geschützten Teile „blockiert" sind, also für die Dauer des Schutzes nur mit Zustimmung des Patentinhabers verwendet werden dürfen.

Grundsätzlich ist der Patentschutz wie alle *Immaterialgüterrechte* zwar territorial begrenzt, d. h. auf das Gebiet desjenigen Landes beschränkt, in dem das Patent erteilt wurde. Durch internationale Abkommen sind allerdings die Patentsysteme der meisten Industriestaaten miteinander verbunden, so dass diese Territorialität relativiert wird. Es ist also relativ leicht, in mehreren Ländern Parallelpatente anzumelden (Hufnagel 2002, S. 280). Eine vollständige Abdeckung aller Staaten, in denen das Internet zugänglich ist, wird man dennoch nicht erreichen. Hinzu kommt, dass ein beträchtlicher Teil der Software über das Netz vertrieben wird. So entsteht die Gefahr unbewusster Patentverletzungen in Staaten, an die der Entwickler u. U. nicht gedacht hat.

Auch inhaltlich ist der Patentschutz vergleichsweise weit. Während das Urheberrecht nur die konkrete sprachliche Ausdrucksform eines Programms, nämlich den Quellcode selbst schützt, betrifft das Patentrecht die im Programm enthaltene technische Lehre in ihrer Gesamtheit, inklusive gleichwirkender Abweichungen im Quelltext. Geschützt ist hier die Funktionalität des Programms, nicht der konkrete Ausdruck dieser Funktionalität. Die Weite des Patentschutzes führt zu verschiedenen Risiken, die die Open-Source-Entwicklung tatsächlich hemmen können (Wiebe 2004*b*).

Lizenzpflicht für die Entwicklung

Besteht die grundsätzliche Möglichkeit, Software auch bei geringer Erfindungshöhe patentieren zu lassen, so kann dies dazu führen, dass geschäftstüchtige Entwickler (Unternehmen) sich schon kleinste Entwicklungsschritte schützen lassen. Auf diese

13 Eine eingehende Analyse des Konfliktpotenzials findet sich bei Wiebe (2004*a*) und Wiebe (2004*b*).

Weise entsteht eine Blockade, die nur von großen Unternehmen mit entsprechendem finanziellen Hintergrund überwunden werden kann.

Probleme bei der Recherche

Hinzu kommt, dass gerade für kleinere Entwickler und Unternehmen eine umfangreiche Patentrecherche schwierig und teuer werden kann, so dass schon das Einholen der Lizenzen ein unüberwindliches Hindernis darstellen kann. Je kleiner das Patent, desto umfangreicher und auch teurer ist die Recherche. Umso schwieriger wird die Nachforschung, weil der Quellcode nach derzeitiger Rechtslage nicht zur *Erfindungsbeschreibung*, also dem Text, der den Umfang des Schutzes definiert, gehört. Die konkreten Anweisungen des Programms sind also nicht erkennbar, so dass alles unter den Schutz fällt, das den gleichen Ablauf hervorruft. Eine „wasserdichte" Patentrecherche ist demnach kaum möglich, wenn der Entwickler nicht bei einem größeren Unternehmen angestellt ist. Daneben dürfte der hohe Zeitaufwand ein Entwicklungshindernis darstellen.

Probleme bei der unternehmerischen Patentrechtspolitik von Open-Source-Unternehmen

Angesichts der zunehmenden Patentierung von Softwarebausteinen insbesondere in den USA sind Praktiken wie die Errichtung von Patentpools und das so genannte *cross-licensing*, bei denen sich Unternehmen ihre Patente wechselseitig lizenzieren, weit verbreitet und finden zunehmend mehr Anerkennung. Eine Teilnahme an diesen Mechanismen ist aber nur möglich, wenn eigene Patente mit eingebracht werden können. Open-Source-Entwickler melden, meist wegen der grundsätzlichen Ablehnung von Patenten in der Präambel der GPL, in der Regel keine Patente an. Da sie deshalb auch keine eigenen Rechte einbringen können, können sie keinen Nutzen aus diesen Formen der Zusammenarbeit ziehen. Im Gegenteil werden sie von der Nutzung zahlreicher zum Teil sehr allgemeingültiger Softwarebausteine ausgeschlossen, die in solchen Pools enthalten oder im Wege des *cross-licensing* freigegeben werden. Dadurch wird nicht nur das Risiko einer Patentverletzung größer, sondern auch der freie Ablauf der Entwicklung gehemmt.

Patentverletzung durch Open-Source-Entwickler

Die systembedingte Offenheit des Quellcodes kann sich schon grundsätzlich als großer Nachteil im Hinblick auf Verletzungen geistigen Eigentums an Software erweisen.[14] Denn während bei Open-Source-Programmen Patent- sowie Urheberrechtsverletzungen einfach erkannt werden können, da über den Quellcode leicht Einblick in die Strukturen des Programms genommen werden kann, ist eine derartige „Patentrecherche" bei nur im Binärcode vertriebenen Programmen nicht möglich.

14 Vgl. dazu ausführlich Lutterbeck und Gehring (2004) und Metzger (2000).

Aufdeckung und Nachweis einer Rechtsverletzung sind bei Open Source deshalb sehr einfach. Das gilt nicht nur für vorsätzliche Verletzungen, sondern auch für unbeabsichtigte und sogar unbewusste. Proprietäre Software ist also doppelt im Vorteil: Weil der Quellcode geheim gehalten wird und eine Dekompilierung zur Aufdeckung einer Patentverletzung nach §§ 69c iVm § 69e Abs. 1 UrhG verboten ist, ist der Nachweis einer Patentverletzung nur schwer oder gar nicht möglich. Andersherum können aber die Inhaber von Patenten an proprietärer Software einfach feststellen, ob ein freies Programm geistiges Eigentum verletzt, da sie Einblick in den Quellcode haben, den sie selbst Dritten verwehren. Für einen Großteil der Open-Source-Entwickler kann diese Tatsache existenzbedrohend sein, da sie häufig unentgeltlich und in ihrer Freizeit arbeiten, sich aber kostenpflichtigen Abmahnungen und Prozessen ausgesetzt sehen, denen sie nichts entgegensetzen können (Nitschke 2002).

Patentanmeldung durch kommerzielle Unternehmen

Patentanmeldungen an im Wesentlichen aus OS-Bestandteilen bestehender Software durch kommerziell ausgerichtete Unternehmen können ebenfalls zu Konflikten führen. Einerseits können auf diese Weise „versteckte" Lizenzen an einem Programm entstehen, die deshalb gefährlich sind, weil sie zu Ansprüchen gegen alle Mitglieder der Vertriebskette führen können. Zudem erlangen die Inhaber der Patente die Macht über den Fortgang des Entwicklungsprozesses, weil sie durch ihre Lizenzpolitik die Verwendung einzelner Bestandteile verhindern können.

Patentanmeldung durch Open-Source-Entwickler

Aber auch aus den „eigenen Reihen" droht Gefahr. Die GPL v2 spricht sich zwar in der Präambel gegen Softwarepatente aus, hindert einen Entwickler aber dennoch nicht daran, ein Patent auf seine Software anzumelden und anschließend andere Nutzer patentrechtlich in Anspruch zu nehmen. Im Rahmen der GPL v2 wird deshalb angenommen, dass die Lizenz eine implizite, also nicht ausdrückliche, *Nutzungsrechtseinräumung* an dem Patent mit umfasst. Das ist deshalb problematisch, weil eine solche konkludente Lizenzerteilung nicht in allen europäischen Mitgliedstaaten anerkannt ist[15] und weil hierdurch u. U. in rechtswidriger Weise in das Recht des Patentinhabers eingegriffen wird, über den Umfang der Lizenzierung selbst zu bestimmen.

4 Abhilfeversuche in der GPL v3

Die neue Lizenz will diesen Schwierigkeiten so weit wie möglich begegnen und den Umgang mit Patenten für Softwarenutzer bzw. -entwickler erleichtern (Jaeger 2006). Im Gegensatz zur GPL v2, die nach allgemeiner Auffassung nur eine implizite Mitlizenzierung der dem Urheber zustehenden Patente enthielt (siehe FSF 2006, S. 3) und

15 Dieses Problem haben auch die Entwickler erkannt, vgl. FSF (2006, S. 17).

sich ansonsten über den Umgang mit möglicherweise am Programm bestehenden Patenten ausschwieg, beinhaltet die GPL v3 mehrere Regelungen, die das Verhältnis der freien Programme zu Softwarepatenten betreffen: Die aktuelle Fassung enthält vor allem einen Verzicht auf die Geltendmachung von Patentansprüchen durch den Weitergebenden, soweit sie diesem zustehen.

Darüber hinaus muss jeder, der GPL-Software weitergeben will und weiß, dass an ihr Patentrechte Dritter bestehen, seine Abnehmer entweder vor der Inanspruchnahme durch den Patentinhaber schützen oder sicherstellen, dass bei der Veröffentlichung des Codes die relevanten Informationen mitgeliefert werden. Dadurch ist er gezwungen, die notwendigen Informationen bezüglich der bestehenden Patente zu veröffentlichen. Ob es stets möglich ist, den Quellcode auf einem öffentlichen Server bereitzustellen, wenn an ihm Schutzrechte bestehen, ist in Zweifel zu ziehen. Wie der Schutz Dritter rechtlich konstruiert werden kann, so dass den Interessen aller Beteiligten wirksam gedient ist, wird ebenfalls Gegenstand der juristischen Diskussion sein müssen.

Es ist vorgesehen, dass jeder Empfänger von GPL-Software automatisch ein einfaches patentrechtliches Nutzungsrecht eingeräumt bekommt. Die erste GPL-Fassung enthielt noch ein solches Nutzungsrecht in ausdrücklicher Form. Da man aber nicht zu stark in die Rechte des Rechteinhabers eingreifen wollte, hat man die ausdrückliche Klausel wieder entfernt. Von einer impliziten Lizenz ist aber weiterhin auszugehen.[16] Die neue Ziffer 2 enthält zudem die Auflage, dass der Nutzer keine *Patentansprüche* bezüglich des betreffenden Programms geltend macht. Anderenfalls verliert er seine Rechte aus der Lizenz, so, wie das in der GPL v2 schon bislang der Fall war, wenn gegen ihre Bedingungen verstoßen wurde. Auf diese Weise soll unter anderem verhindert werden, dass Entwickler eigene Fortentwicklungen „heimlich" patentieren lassen und dann andere Nutzer oder Entwickler, die gleichwertige Änderungen vornehmen, rechtlich verfolgen. Der *Rechterückfall* war in Bezug auf die Verletzung der urheberrechtlichen Nutzungsbefugnisse schon in der zweiten Version umstritten.[17]

Daneben lässt die GPL v3 eine Kombination mit Lizenzen zu, die noch weitergehende *Patentkonterklauseln* enthalten. Bemerkenswert ist hier, dass sich die Pauschalkritik an Softwarepatenten laut Präambel nun ausschließlich auf Patentierungen von Software für *general-purpose computers* bezieht. Hardwareimplementierte Software wie beispielsweise die Firmware eines Kernspintomographen, die eine fest verbundene Einheit mit der Hardware bildet, soll auch nach Auffassung der GPL-Entwickler patentierbar sein.

5 Wirksames Heilmittel oder nur Symptomtherapie?

Sind aber nun die Änderungen eine wirksame Arznei gegen die durch die Patentierung von Computerprogrammen entstehenden Gefahren oder bekämpfen sie nur

16 Vgl. dazu mit http://www.fsf.org/news/gplv3-clarification.

17 Dazu statt vieler: Schiffner (2002), Spindler (2004) und Jaeger und Metzger (2006).

die Symptome, ohne auf das eigentliche Problem einzugehen? Da ist zum einen der Verzicht auf die Geltendmachung von Patentansprüchen durch denjenigen, der die Software weiterverbreitet. Hier stellt sich eine praktische Schwierigkeit, die aus juristischen Feinheiten resultiert. Patente bleiben nämlich nicht immer in der Hand des Erfinders, sondern können auch wie eine Sache verkauft werden. Diese Tatsache kann und wird im Zusammenhang mit dem Verzicht auf die Geltendmachung von Patentansprüchen zu Schwierigkeiten führen. Ein solcher Verzicht kann nämlich nur schuldrechtlich wirken, d. h. nur zwischen den Parteien, die ihn tatsächlich vereinbart haben, nicht gegenüber jedermann. Wird ein Patent also an jemand anderen verkauft, ist der Käufer dem Lizenznehmer gegenüber nicht mehr verpflichtet, auf die Geltendmachung von etwaigen Verletzungsansprüchen zu verzichten und kann diesen gegebenenfalls patentrechtlich verfolgen. Allein der Verzicht auf die Geltendmachung von Patentansprüchen reicht also nicht aus, um einen wirkungsvollen Schutz des Abnehmers zu gewährleisten.

Das kann schon eher durch die Verpflichtung zum Schutz der Abnehmer vor Patentansprüchen gelingen,[18] deren juristische Konstruktion allerdings wie bereits angesprochen ungeklärt ist. In Betracht kommt erst einmal eine *befreiende Schuldübernahme*, bei der sich der Weitergebende seinem Abnehmer gegenüber verpflichtet, eine bei diesem entstehende Schuld vollständig zu übernehmen. Diese Konstruktion hat den Nachteil, dass sie gemäß § 415 BGB vom Patentinhaber als Gläubiger genehmigt werden muss, dessen Feststellung gerade bei Open-Source-Software manchmal recht aufwändig sein kann. Einfacher wäre ein „Schuldbeitritt mit Freistellungsverpflichtung", bei dem der Weitergebende der Schuld des Patentverletzers beitritt, also mit haftet, und im Innenverhältnis, also nur im Verhältnis zwischen den beiden, verpflichtet ist, seinen Abnehmer freizustellen. Zwischen beiden entsteht eine *Gesamtschuld*, es können also beide vom Patentinhaber in Anspruch genommen werden. Der Empfänger der Software kann sich sein Geld von dem Weitergebenden zurückholen, umgekehrt aber nicht. Daneben kann man auch noch annehmen, Distributor und Empfänger schlössen einen „konkludenten Garantievertrag" ab. Dabei gibt der Weitergebende, ohne es ausdrücklich zu sagen, eine Garantie dafür ab, dass der Nutzer nicht für eventuelle Patentverletzungen haften muss. Wird er dann doch in Anspruch genommen, so ist der Weitergebende auf Grund des Garantievertrages verpflichtet, den Betrag zu übernehmen. Welche dieser Konstruktionen sich letztendlich durchsetzen kann, wird stark von den jeweiligen Einzelverhältnissen abhängen. Dieses Schutzinstrument scheint zumindest in Bezug auf solche Patentansprüche, von denen der Distributor weiß, recht wirkungsvoll. Es kann aber nicht davon ausgegangen werden, dass auf diese Weise vollständige Sicherheit des Empfängers vor der Inanspruchnahme von Patentinhabern gewährleistet werden kann. Denn einerseits muss damit gerechnet

18 Die praktische Bedeutung dieser Verpflichtung muss sich allerdings erst noch zeigen. Die Möglichkeit, die notwendigen Informationen auf den Server zu laden, auf dem die Software angeboten wird, stellt einen erheblich geringeren Aufwand dar und wird deshalb vielleicht mehr Erfolg haben.

werden, dass es Patente gibt, von denen der Distributor nichts weiß und andererseits ist er nicht gezwungen, sich auf diese Auflage einzulassen. Er kann stattdessen auch den Code auf einem öffentlichen Server zur Verfügung stellen und damit die notwendigen Informationen über gegebenenfalls bestehende Patente veröffentlichen. Das wiederum wird ohne Patentverletzung nicht immer möglich sein, so dass auch hier kein vollständig wirksamer Schutz des Abnehmers gewährleistet werden kann.

Die Möglichkeit einer konkludenten Lizenzerteilung betrifft wiederum nur solche Nutzungsrechte, hinsichtlich derer der Distributor verfügungsbefugt ist. Er kann keine Lizenz erteilen für solche Patente, die jemand anderem gehören. Juristisch stellt es sich auch als problematisch dar, ob man eine konkludente Lizenz, die sich allenfalls aus einer Auslegung der konkreten Vereinbarung ergeben kann, neben einer ausdrücklichen Regelung wie der neuen Ziff. 11 tatsächlich wird annehmen können. Hinzu kommt, dass die Wirksamkeit einer solchen impliziten Lizenz im europäischen Rechtsraum zum Teil fraglich ist. Die Möglichkeit einer konkludenten Lizenzerteilung schadet also nicht, hilft aber auch nicht unbedingt gegen die Risiken von Softwarepatenten.

Am wirkungsvollsten, aber juristisch zugleich auch am schwierigsten zu beurteilen ist die Regelung, nach welcher der Nutzer alle Rechte aus der GPL verliert, wenn er *Patentansprüche* bezüglich der lizenzierten Software geltend macht. Mit dieser Klausel soll die heimliche Patentierung vermieden werden. Sie erinnert stark an die GPL v2, in der sich der Rechtewegfall auf eine Verletzung der urheberrechtlichen Bedingungen der Lizenz stützte. Schon hier waren die rechtliche Konstruktion und die Wirksamkeit eines solchen, „dinglichen", also gegenüber jedermann wirksamen, Rechtsverlustes ungeklärt. Letztendlich bestehen hier keine Unterschiede im Vergleich zu der ursprünglichen Regelung, die von der herrschenden Meinung als auflösend bedingte *Nutzungsrechtseinräumung* angesehen wurde. Auch die Geltendmachung von Patentansprüchen stellt eine auflösende Bedingung hinsichtlich der durch die GPL eingeräumten Rechte dar. Insofern kann auf die Diskussion zur GPL v2 verwiesen werden, wonach die Wirksamkeit dieser Konstruktion umstritten ist.[19] Endgültig sicherer Schutz vor heimlicher Patentierung durch Open-Source-Entwickler kann damit durch diese Regelung nicht gewährleistet sein.

Die erweiterte Kompatibilität der GPL, die nunmehr auch eine Kombination mit Software zulässt, die unter einer Lizenz mit mehr Möglichkeiten zu *Patentkontern* steht, hat letztlich auf die Sicherheit und den Schutz vor den genannten Risiken keinerlei Auswirkungen und verdeutlicht nur, dass die Verfasser der GPL Softwarepatenten nach wie vor kritisch gegenüberstehen.

6 Fazit

Tatsächlich birgt die Patentierung von Software einige Risiken für Open-Source–Entwickler und -Anwender. Zu nennen ist insbesondere die schwierige Recherche,

19 Nachweise siehe oben unter Abschnitt 4.

während gleichzeitig Patentverletzungen durch Open-Source-Entwickler durch den offen zugänglichen Quellcode leicht erkennbar sind. Gegen die besonders in den USA verwurzelte Praxis der Patentrechtspools kann die GPL nur insofern helfen, als sie die Patentierung auch von freier Software ausdrücklich zulässt, aber verlangt, dass jedem Nutzer eine unentgeltliche Lizenz an den betreffenden Patenten eingeräumt wird. Ob dadurch ein in Patentpools nützliches Zahlungsmittel entstehen kann, lässt sich allerdings auf einer rein theoretischen Ebene nur schwer beantworten.

Durch die Recherchehindernisse entsteht ein weiteres Problem, nämlich das der Patentverletzung durch Open-Source-Entwickler. Da es schwierig ist, herauszufinden, welche Codebestandteile schon patentiert sind, erhöht sich natürlich das Risiko, beim Entwickeln eine Patentverletzung zu begehen. Diesem Risiko können auch die Neuregelungen in der GPL v3 nicht abhelfen, da der Distributor nur insoweit Schutz gewähren kann und muss, als er von den bestehenden Patentansprüchen weiß. Ist dies nicht der Fall, so gibt es auch keinen Schutz.

Zumindest eine abschreckende Wirkung werden die Neuregelungen aber auf solche Entwickler freier Software haben, die sich selbst mit dem Gedanken tragen, ihre Entwicklungen patentieren zu lassen. Das ist zwar grundsätzlich möglich, die Gefahr, die daraus für andere Entwickler entsteht, wird jedoch durch die Regelungen in der neuen GPL gebremst. Wer solche Software weitergibt, muss darauf verzichten, die ihm zustehenden *Patentansprüche* gegenüber den Nutzern und Weiterentwicklern des Programms geltend zu machen, wenn er es unter der GPL lizenzieren will. Ebenso kann sich aus den Umständen eine konkludent erteilte einfache Lizenz ergeben. Hinzu kommt, dass jeder Distributor seine Abnehmer vor solchen Patentansprüchen schützen muss, von denen er weiß, also auch vor seinen eigenen. Tut er das nicht, so muss er die Informationen über bestehende Patente und den Code veröffentlichen, so dass auch hier keine Unsicherheit mehr bestehen kann.

Im Ergebnis lässt sich also festhalten, dass die Neuregelungen in der GPL v3 zwar wirkungsvoll vor der Gefahr aus den eigenen Reihen schützen, das eigentliche Problem, nämlich das Dilemma zwischen schwieriger Patentrecherche bei proprietärer Software und dem offenen Code können sie jedoch nicht vollständig lösen. Das wird letztendlich Aufgabe des Gesetzgebers sein, der ein Patentrecht schaffen muss, das kommerzielle und Open-Source-Interessen in Einklang bringt.[20]

Literatur

Busse, R. (2003), *Patentgesetz, Kommentar*, 6. Aufl., de Gruyter, Berlin.

FSF (2006), 'GPLv3 First Discussion Draft Rationale', Free Software Foundation. http://gplv3.fsf.org/gpl-rationale-2006-01-16.pdf [12. Jan 2007].

20 Zu den rechtspolitischen Optionen siehe Horns (2000). Vorschläge gibt es auch bei Weyand und Haase (2004, S. 203) und bei Lutterbeck und Gehring (2004, S. 12 ff.). Die ausführliche Diskussion dieser Vorschläge würde den Rahmen dieses Beitrags allerdings bei Weitem sprengen.

FSFE (2006), 'Patents and GPLv3', Free Software Foundation Europe. http://www.germany.fsfeurope.org/projects/gplv3/patents-and-gplv3.en.html [02. Jan 2007].

Horns, A. H. (2000), 'Der Patentschutz für softwarebezogene Erfindungen im Verhältnis zu „Open Source"-Software', JurPC. http://www.jurpc.de/aufsatz/20000223.htm.

Hufnagel, F. (2002), 'Software- und Business-Patente – Herausforderung für das juristische Risikomanagement', *Multimedia und Recht* (5), S. 279 ff.

Jaeger, T. (2006), 'Eine Lizenz entsteht – Neue Entwürfe für GPL und LGPL', heise open. http://www.heise.de/open/artikel/76248 [12. Dez 2006].

Jaeger, T. und Metzger, A. (2006), *Open Source Software. Rechtliche Rahmenbedingungen der Freien Software*, 2. Aufl., C. H. Beck, München.

Klopmeier, F. (2002), 'Zur Technizität von Software', *Mitteilungen der deutschen Patentanwälte* **93**(2), S. 65–70.

Lutterbeck, B. und Gehring, R. A. (2004), Software-Patente im Spiegel von Softwareentwicklung und Open Source Software, *in* J. Taeger und A. Wiebe (Hrsg.), 'Informatik – Wirtschaft – Recht: Regulierung in der Wissensgesellschaft, Festschrift für Wolgang Kilian', Nomos, Baden-Baden, S. 301–322. http://ig.cs.tu-berlin.de/ma/rg/ap/2003-x/GehringLutterbeck-SWPat-092003.pdf [02. Jan 2007].

Metzger, A. (2000), 'Softwarepatente als Gefahr', *Linux Magazin* (7), S. 64–65. http://www.ifross.de/ifross_html/art4.html [02. Jan 2007].

Metzger, A. (2003), 'Softwarepatente im künftigen europäischen Patentrecht', *Computer und Recht* **19**(5), S. 313–317. http://www.ifross.de/ifross_html/art45.pdf [02. Jan 2007].

Nack, R. (2004), 'Neue Gedanken zur Patentierbarkeit von computerimplementierten Erfindungen: Bedenken gegen Softwarepatente – ein déjà vu?', *GRUR Int* (9), S. 771–775.

Nitschke, T. (2002), 'Geistiges Eigentum ist Diebstahl. Freie Software vs. Software-Patente', *Forum Recht Online* (3), S. 91–93. http://www.forum-recht-online.de/2002/302/302nitschke.htm [02. Jan 2007].

Pierson, M. (2004), 'Softwarepatente – Meilensteine der patentrechtlichen Rechtsprechung', JurPC. http://www.jurpc.com/aufsatz/20040182.htm [13. Jan 2007].

Schiffner, T. (2002), *Open Source Software. Freie Software im deutschen Urheber- und Vertragsrecht*, 1. Aufl., VVF, München.

Spindler, G. (2004), *Rechtsfragen bei Open Source*, Otto Schmidt, Köln.

Weyand, J. und Haase, H. (2004), 'Anforderungen an einen Patentschutz für Computerprogramme', *GRUR Int* (3), S. 198 ff.

Wiebe, A. (2004*a*), 'Softwarepatente – das Ende von Open Source?', *Medien und Recht* (3), S. 195 ff.

Wiebe, A. (2004*b*), 'Softwarepatente und Open Source', *Computer und Recht* **20**(12), S. 881–888.