Praxishandbuch Open Source

Technische und rechtliche Rahmenbedingungen für einen lizenzkonformen Einsatz von FOSS im Unternehmen

Herausgegeben von

Christian Galetzka, LL.M.
Rechtsanwalt und Fachanwalt für IT-Recht, Würzburg

Chan-jo Jun
Rechtsanwalt und Fachanwalt für IT-Recht, Würzburg

und

Yvonne Roßmann
Rechtsanwältin und Fachanwältin für IT-Recht, Würzburg

Bearbeitet von

Lisa Breunung; Christian Galetzka, LL.M.; Florian Hackel;
Chan-jo Jun; Julia Kendziorra; Ulrich Kulke; Yvonne Roßmann

Fachmedien Recht und Wirtschaft | dfv Mediengruppe | Frankfurt am Main

Bibliografische Information der Deutschen Nationalbibliothek

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.de abrufbar.

ISBN 978-3-8005-1763-3

dfv Mediengruppe

Klimaneutraler Druck in Deutschland

www.ruw.de

Druck: WIRmachenDRUCK GmbH, Backnang

Printed in Germany

Kapitel I
Free and Open Source Software (FOSS) – Idee und Risiken

Der technische Einsatz von Free and Open Source Software (FOSS) ist weit verbreitet, das gilt jedoch nicht im gleichen Umfang für die rechtliche Einhaltung der Lizenzvorgaben. Hier treffen zwei Welten aufeinander, die allen Beteuerungen zum Trotz lieber nebeneinander koexistieren möchten, als sich allzu weit zu überschneiden. Software-Entwickler möchten die langen Lizenztexte am liebsten auf eine einzeilige If-then-else-Formel reduzieren und Juristen wollen für die Lizenzbedingungen keine Software-Architektur analysieren. **1**

Basic: Free and Open Source Software (FOSS) (siehe Rn. 63 ff.) **2**

Die wesentlichen Merkmale von FOSS sind, dass die Software von jedermann frei und ohne Zahlung von Lizenzgebühren verwendet werden kann, der Source Code der Software zugänglich ist und der Nutzer die Software selbst verändern und an Dritte weitergeben darf. Freeware, Shareware und auch Public Domain Software sind keine FOSS im klassischen Sinne; letztere kann jedoch nach denselben Maßstäben beurteilt werden.

FOSS Komponenten stehen regelmäßig unter klassischen FOSS Lizenzen, die von der FSF[1] oder der OSI[2] nach deren Kriterien anerkannt wurden (zu diesen Organisationen siehe Rn. 50 ff.). Sie lassen sich meist der bei der Organisation SDPX verfügbaren Lizenzliste entnehmen.[3] Wir verwenden in diesem Buch für alle FOSS Lizenzen als Bezeichnungen die in dieser Liste vorgesehenen SPDX Identifier, also die vereinheitlichten Kurzbezeichnungen. SPDX ist ein Projekt, das sich die einheitliche Bezeichnung und Übermittlung von FOSS nach einheitlichen Standards auf die Fahne geschrieben hat.[4]

Man kann sich damit begnügen, alle GPL-artigen Lizenzen auf eine rote Liste zu setzen und schlicht nach passenden Komponenten un- **3**

1 https://www.gnu.org/licenses/license-list.html.en.
2 https://opensource.org/licenses.
3 https://spdx.org/licenses/.
4 https://spdx.dev/.

ter liberalen Lizenzen zu suchen. Dabei werden die Entwickler jedoch schnell feststellen, dass gerade die häufig benötigten, gewollten oder schlicht am besten geeigneten Komponenten unter „gefährlicheren" Lizenzen stehen. Diese können häufig nur eingesetzt werden, wenn bei der Verwendung genaue technische Vorgaben eingehalten werden. Insofern ist es für eine vertiefte Auseinandersetzung unerlässlich, entsprechendes Technikwissen zu haben. Damit lässt sich ein Einsatz oft lizenzkonform gestalten, auch wenn die erste rechtliche Bewertung zunächst ein höheres Risiko ausgibt. Um solche Fälle zu beurteilen, braucht man technische und rechtliche Grundlagen, die ein wenig über das Allgemeinwissen hinausgehen.

4 Die Schnittstelle zwischen den Juristen und Entwicklern ist dabei oft die Sollbruchstelle. Fähige Entwickler können den Code für Lizenzkonformität anpassen, wenn sie wissen, welche Software-Architektur den Lizenzvorgaben gerecht wird. Gute Juristen können Lösungen für technisch nicht anders umsetzbare Architekturen finden, wenn sie die dahinter liegenden Code Mechanismen, beispielsweise zur Interaktion verschiedener Komponenten miteinander, verstanden haben. In den folgenden Kapiteln dieses Buches finden beide Parteien die entsprechenden Grundlagen aus dem rechtlichen und technischen Bereich praxisnah aufbereitet. Ziel ist, das Zusammenspiel von Technik und Recht zu optimieren und so den lizenzgerechten Einsatz von FOSS zu gestalten.

1. Wie naiver Einsatz kostenloser Software Ihr Unternehmen bedrohen kann

5

- Die FOSS Idealisten wollten die Software-Entwicklung von den Beschränkungen des Urheberrechts befreien, knüpften die Freiheit aber an Bedingungen.
- Alternativlosigkeit von FOSS wird schnell behauptet, sollte aber kritisch hinterfragt werden.
- Vor- und Nachteile sollten bei einem Einsatz von FOSS gut abgewogen werden, insbesondere wenn es kommerziell lizenzierte Alternativen gibt.

– Neben Rufschäden drohen auch gerichtliche Verfahren bei FOSS Lizenzverletzungen.

Welche Motivation steckt hinter dem FOSS Konzept? Die urheber- und patentrechtlichen Beschränkungen von kommerzieller Software stellen sich als Hinderungsgrund für die kreative Entfaltung dar. Hätten nicht alle mehr davon, wenn jeder auf die bestehende Software aufsetzen könnte und dann seine eigenen Ergebnisse ebenfalls wieder im Source Code bereitstellt? Die Grundidee war einfach und altruistisch. Der Idealismus stößt jedoch an Grenzen, wenn sich zwar alle bedienen wollen, wichtige Fortentwicklungen dann jedoch unter kommerziellen Lizenzen verwertet wurden. So wurde die gegenseitige Freizügigkeit zur einklagbaren Bedingung. **6**

a) Am Anfang stand die Idee: Freiheit von Copyrightzwängen

Der Entwickler *Richard Stallman* war einer der ersten, der die Idee von FOSS prägte. Zur Erläuterung verwies er darauf, dass das Konzept „frei" i. S. v. Freiheiten, nicht i. S. v. nur kostenfrei bedeuten soll. **7**

> *„To understand the concept, you should think of ‚free' as in ‚free speech', not as in ‚free beer'."*

ist das wesentliche, von *Stallman* entwickelte Leitmotiv der FOSS Community.[5] Mit verfügbarem Source Code könnten alle Entwickler an der Lösung aller Probleme und für alle benötigten Funktionen weltweit zusammenarbeiten, ohne dass jemand Wissen für sich behält.

aa) Wie sich FOSS verbreitet hat

Von diesem Ideal ist die heutige Realität jedoch deutlich entfernt. Der Gedanke, Geld für Software-Entwicklung auszugeben und die Ergebnisse dann an die Konkurrenz zu verschenken, erscheint in vielen Branchen noch immer einigermaßen abwegig. Hinzu kommt die Sorge vor der Aufdeckung von Sicherheitslücken, wenn Code offengelegt **8**

5 Zu den wesentlichen Grundprinzipien Freier Software *Stallman*, https://www.gnu.org/philosophy/free-sw.html.en.

wird. Zwar wird viel FOSS eingesetzt, häufig aber eingebettet in kommerziellen Code.

9 Zur Jahrtausendwende war FOSS Software schon weit verbreitet, für manche seriösen Anwender aber gleichwohl anrüchig. Software-Anwender wollten einen Lizenzgeber haben, den sie in Haftung nehmen können, wenn etwas an der Software nicht funktioniert. Nur Freaks, die sich von Pizza und Cola ernährten, betrieben Linux Server zuhause, während seriöse Unternehmen mit viel Mühe Windows NT-Server am Laufen hielten. Die Entscheidung für FOSS Anwendungen war nicht alternativlos, erfolgte häufig aus einer bewussten Entscheidung zugunsten der Quelloffenheit und im Konsens mit dem Grundprinzip, dass man im Gegenzug zur kostenlosen Zurverfügungstellung der Software auch seine Bearbeitungen an die Community zurückspielen werde. In den folgenden Dekaden entstanden eine Kommerzialisierung von FOSS und eine Entkopplung von diesem altruistischen Grundverständnis.

10 Heutige Software-Entwickler beginnen ihren Beruf mit der Überzeugung, dass die Mehrheit aller Entwicklungsaufgaben unter Einbindung einer FOSS Komponente gelöst werden könne. Der Gedanke, auf FOSS zu verzichten und stattdessen alles von Hand zu programmieren, erscheint so absurd wie der Vorschlag an den Bäcker, sein eigenes Mehl zu mahlen. Während die Open Source Communities zwar gewachsen sind, wuchs aber auch der Anteil der kommerziellen Anwender, die unter keinen Umständen ihre eigenen Software-Quellen offen bereitstellen würden. Diese unterschiedlichen Einstellungen spielen eine Rolle bei der Auslegung der Lizenzen, wenn der Wortlaut nicht ausreicht und auf den mutmaßlichen Willen des Lizenzgebers abgestellt werden soll. Hier wird in beide Richtungen leidenschaftlich argumentiert, dass die FOSS Lizenz entweder jegliche kommerzielle Nutzung ohne unüberwindbare Hindernisse zulassen wollte oder anderseits der vollständige Support für einen Austausch gewährleistet sein muss.

bb) Wo FOSS verzichtbar ist und wo nicht

11 Software-Entwickler haben gelegentlich Mühe, ihren Inhouse-Juristen zu erklären, dass ein kompletter Verzicht von FOSS genauso wenig denkbar ist wie der Vorschlag, auf sämtliche Copyleft Lizenzen

zu verzichten. Wer sich auf ein Linux Ökosystem eingelassen hat, kann den Einsatz von GPL nicht kategorisch ausschließen. Das einfache Modell, Lizenzen in Gut und Böse einzuteilen, hält dem Realitätscheck nicht lange stand. Wohl dem, der bei Entwicklung seiner Lizenzrichtlinien rechtzeitig mit seinen Entwicklern gesprochen hat.

Basic: Copyleft (siehe Rn. 225 ff.) **12**

Das ist das zentrale Prinzip des FOSS Ideals. Hat die Lizenz ein Copyleft, so muss ein von der FOSS Komponente „abgeleitetes Werk" ebenfalls unter den Bedingungen der FOSS Lizenz verfügbar gemacht werden. Es gibt Lizenzen mit strengem Copyleft, die dieses Prinzip voll umsetzen, andere mit beschränktem Copyleft, die teils Ausnahmen vorsehen, sowie Lizenzen ohne Copyleft, die nur die Einhaltung anderer Bedingungen vorsehen. In der Regel wollen Unternehmen Copyleft vermeiden, um eigenen bzw. proprietären Code nicht im Source Code unter einer FOSS Lizenz freigeben zu müssen, damit sie die Lizenzvorgaben einhalten.

Auf der anderen Seite wird das Argument der technischen Notwendigkeit und Alternativlosigkeit viel zu häufig hingenommen. Wer sich dann im nächsten Schritt auch sagen lässt, dass die Einhaltung der theoretischen Bedingungen praktisch oder wirtschaftlich unmöglich ist, findet sich im Dilemma, zwischen Lizenzverstoß und Lästigkeit entscheiden zu müssen. Man solle doch pragmatisch sein, statt Probleme zu schaffen, wettert der CTO gegenüber der Rechtsabteilung und diese antwortet dann mit einer Strafbarkeitswarnung an den CEO. Diese Eskalation ist am Ende peinlich, wenn sich später herausstellt, dass eine legale Lösung nur aus Unfähigkeit übersehen wurde. **13**

Zugegeben, an FOSS kommt man kaum vorbei, sobald ein Stromverbraucher mehr macht, als einen Wolfram-Draht zu erwärmen. Früher konnte man die typischen Einsatzfelder von FOSS noch mit Linux Servern beschreiben, während eingebettete Software häufiger individuell programmierte Software enthielt. Mit dem Internet der Dinge ist kaum noch eine Anwendung zu primitiv, um nicht mit FOSS umgesetzt zu werden. Eine Glühbirne kommt heute mit WLAN-Empfang auf FOSS Basis und synchronisiert das Programm mit Heizung, Ka- **14**

mera und Lautsprecher, ebenso das ferngesteuerte Sexspielzeug. Was kommuniziert, enthält vermutlich auch FOSS.

15 Kommerzielle Betriebssysteme wie iOS oder Windows enthalten bereits selbst FOSS und die dafür geschriebenen Anwendungsprogramme erst recht. Selbst die Steuerung einer Sieben-Segment-Anzeige wird gelegentlich mit FOSS realisiert. Wenn man FOSS freie Systeme sucht, müsste man daher schon auf die analoge Ebene des Operationsverstärkers zurückgehen – der enthält garantiert keine FOSS.

16 Es gibt jedoch auch leistungsfähige Realtime Betriebssysteme, die ohne Copyleft Lizenzen auskommen, dafür aber Lizenzgebühren kosten. Teilweise geben Rechtsinhaber die gleiche Anwendung unter FOSS Lizenz oder unter kommerzieller Lizenz heraus, so dass der Verzicht auf Copyleft Effekt gegen Gebühr gekauft werden kann. Bei vielen Libraries gibt es alternative Komponenten unter liberalen Lizenzen, was manchmal erst spät im Entwicklungsprozess bemerkt wird. Juristen sind daher gut beraten, bei behaupteter Alternativlosigkeit genauer hinzuschauen.

17

Basic: Begriffsklärung Software/Komponente/Produkt/Projekt/Library/File

Im Bereich FOSS werden viele Begriffe unterschiedlich eingesetzt. Wir verwenden als zentrale Bezeichnung FOSS Komponente für das einzelne urheberrechtlich geschützte Werk, das abgrenzbar für die Software-Entwicklung eingesetzt wird. Parallel werden auch die Begriffe FOSS, Produkt oder Projekt verwendet. Gerade die Internetauftritte von FOSS Komponenten werden häufig als Projekthomepages bezeichnet und bei entsprechender Community die Gemeinschaft/Komponente oft als Projekt. Ist die Komponente unselbstständig, wird sie häufig als Library bezeichnet (typisches Format ist eine .dll-Datei).

Eine FOSS Komponente besteht aus einer Vielzahl einzelner Files, die Zeilen von Header Informationen und Code beinhalten. Sie enthält ggf. weitere Komponenten oder Komponentenbestandteile, sogenannte Abhängigkeiten oder Dependencies. Teils sind diese nicht direkt in den Code einbezogen, sondern werden erst beim Kompilierungsvorgang, also der Umwandlung des Source Code in ein Kompilat, hinzugezogen.

Wir setzen den Begriff Produkt nicht für die einzelne Komponente ein, sondern wenn wir das Ergebnis der Zusammenstellung der Komponenten ggf. auch mit proprietärem Code bezeichnen wollen. Also ist damit quasi das Endergebnis der Programmiervorgänge gemeint. Alternativ wird das auch als Projekt bezeichnet. Projekt nutzen wir jedoch als Referenz für den gesamten Entwicklungsvorgang, insbesondere mit Bezug auf die Zeitschienen.

b) In welchen Fällen kostenlose Software teuer werden kann

18 FOSS gewinnt nicht nur Territorium in Wegwerf-Hardware, sondern auch in hochspezialisierten und hochpreisigen Steuergeräten für Maschinen oder Fahrzeuge. Ein Entscheidungsträger, der für die nächste Gerätegeneration einen Wechsel von vorwiegend proprietär lizenzierten Systemen zu FOSS beschließt, kennt die technischen Vorzüge und die ersparten Lizenzkosten sehr genau. Die rechtlichen Kosten treffen einige Hersteller aber trotzdem unvorbereitet und verdienen eine frühere Betrachtung.

19 Viele FOSS Entwickler verkennen zum Zeitpunkt der Systementscheidungen den Entwicklungsaufwand, der für eine lizenzkonforme Umsetzung erforderlich ist. Gerade bei der tiefen Einbindung von Copyleft Komponenten ist die Freigabe von eigenem Source Code manchmal nicht zu verhindern. Wer diese Software nicht freigeben will oder kann, weil sie ihm gar nicht selbst gehört, findet sich in einer Sackgasse und nimmt entweder Rechtsverletzungen in Kauf oder revidiert seine Software-Auswahl.

20 Wenn die gleiche Software unter Dual License gegen Lizenzgebühr oder unter Copyleft Lizenz angeboten wird, entfällt einerseits das Argument der Alternativlosigkeit, andererseits erhöht sich das Risiko der Rechtsverfolgung, da der Rechtsinhaber eher geneigt ist, Verletzungen der FOSS Variante zu verfolgen und Schadensersatz auf Grundlage der Lizenzanalogie einzufordern.

21 Auf der Seite der Schadenspotenziale sollte kalkuliert werden, welche Auswirkungen ein erfolgreich geltend gemachter Unterlassungsanspruch hätte. Ein solcher Anspruch sorgt bis zum Austausch der Software mindestens zu einem Auslieferungsstopp, u. U. auch zu einem Produktrückruf. Eine solche Rückrufpflicht ergibt sich für bereits

veräußerte Gegenstände nicht automatisch aus dem Unterlassungsanspruch, mittelbar jedoch aus den Gewährleistungsansprüchen, wenn das nunmehr illegale Werk vom Käufer nicht mehr weitergegeben werden kann. Der Importeur einer Webcam mag dieses Risiko niedriger einschätzen als ein Automobilhersteller, dem die Stilllegung seiner Flotte droht. Im Business-to-Business-Bereich wiederum können Nachbesserungsaufwände im Rahmen von Wartungsverträgen zu verschmerzen sein.

22 Bei Umsetzung der künftig vorgeschriebenen Over-the-Air-Updates ergeben sich leichtere Korrekturmöglichkeiten für heilbare Lizenzverletzungen wie beispielsweise fehlende Pflichtangaben. Soweit jedoch die Lizenzkonformität nur durch Freigabe von eigenem Source Code erreicht werden kann, kann dies schon alleine daran scheitern, dass der Lizenzverletzer gar nicht über die notwendigen Rechte verfügt, um den entsprechenden Code unter einer FOSS Lizenz freizugeben.

23

Basic: Pflichtangaben und Source Code Freigabe (siehe Rn. 765 ff.)

Die FOSS Lizenzen unterscheiden sich teils deutlich im Regelungsgehalt und Umfang der Regelungen, insbesondere bzgl. des Copyleft. So gut wie alle FOSS Lizenzen fordern jedoch für Verwertungshandlungen eine Auflistung von bestimmten Angaben, wir nennen sie Pflichtangaben. Es handelt sich meist um Copyright Hinweise, Lizenztexte und teils weitere Angaben wie z. B. Informationen zur Veränderung.

Copyleft Lizenzen fordern in der Regel die Mitlieferung oder das Angebot der Mitlieferung des Source Code der FOSS Komponente unter der entsprechenden Lizenz, auch wenn das Copyleft für verbundene oder veränderte Software nicht greift. Greift Copyleft, muss entsprechend auch der veränderte oder verbundene Code bereitgestellt werden.

24 Wer selbst innerhalb einer Zulieferkette FOSS einsetzt, muss in seiner Kalkulation berücksichtigen, dass er für die Schäden seines Kunden haftet, soweit es ihm nicht gelingt, diese Haftungen vertraglich zuverlässig auszuschließen, worauf wir in diesem Buch noch eingehen werden. Natürlich sind die Zeiten vorbei, in denen Rechtsabteilungen apodiktisch jeglichen Einsatz von FOSS verboten haben. Andererseits verbietet sich der kritiklose ungeprüfte Einsatz.

c) Was schlimmstenfalls passieren kann

Die Rechtsabteilungen haben selbstverständlich im Blick, wie wahrscheinlich und groß die drohenden Schadensszenarien sind. Eines ist meist eindeutig klar: Eigener proprietärer Code soll nicht in großem Umfang unter einer FOSS Lizenz freigegeben werden. Die Vorgabe, Copyrightvermerke zusammen mit der jeweiligen Lizenz bei Weitergabe beizufügen, nehmen die Verwender meist schon deutlich weniger ernst. In der Praxis kommt häufig die Frage, ob das denn überhaupt jemand merkt und dann tatsächlich auch noch verfolgt. **25**

aa) Häufiger Irrtum: Copyleft führt automatisch zur Freigabe als FOSS

Die Geschichte ist so dramatisch, dass sie in fast jedem FOSS Vortrag vorkommt. Sie ist trotzdem falsch. Es wird behauptet, dass die Auslösung des Copyleft Effekts dazu führt, dass die eigene Software – infiziert von einer winzigen Dosis Copyleft – auch zum Copyleft Zombie wird. Auf dieser Grundlage wird dann die Freigabe des kompletten kommerziellen Produkts gefordert, das entsprechend infiziert ist. Richtig ist, dass Werke, die von Copyleft lizenzierten Werken abgeleitet werden, ebenfalls unter Copyleft Lizenz gestellt werden müssen, um die Lizenzbedingung zu erfüllen (siehe hierzu Rn. 225 ff.). Es gibt jedoch keinen Automatismus. **26**

Der Rechtsinhaber hat die Wahl, die Copyleft Lizenz dadurch zu erfüllen, dass er seine eigene Software auch unter gleiche Lizenz stellt oder auf die Vorzüge der FOSS Lizenz, insbesondere auf das gewährte Nutzungsrecht zu verzichten. Das mag sich nach einem Dilemma anhören, da die Verletzung der Lizenz Rechtsnachteile bedeuten kann; es ist aber gleichwohl die Entscheidung des Rechtsinhabers, ob er die Freigabe seines Code und der FOSS Lizenz vornehmen will oder nicht. Dass kein Automatismus bestehen kann, ergibt sich schon alleine aus dem Umstand, dass der Verwerter möglicherweise gar nicht in der Lage ist, eine Lizenzierung unter einer FOSS Lizenz vorzunehmen. Wäre es anders, könnte man eine fremde kommerzielle Software mal eben mit GPL infizieren, so dass sie dann für jedermann frei nutzbar wäre. **27**

28 Der Gedanke hält sich gleichwohl hartnäckig und liegt sogar der Entscheidung des LG Berlin im Fall AVM ./. Cybits (Surfsitter)[6] zugrunde. Das an mehreren Stellen angreifbare Urteil (siehe ausführlich Rn. 605 ff.) geht davon aus, dass der Hersteller der Fritzbox keine Unterlassungsansprüche gegen Cybits geltend machen darf, wenn das linux-basierte FritzOS verändert wird. Die Richter statuierten, dass mit Copyleft Effekt infizierte Software quasi „vogelfrei" wird und der Rechtsinhaber seine Ansprüche aus dem Urheberrecht verlieren soll. Anders ausgedrückt sollen Sammelwerke, die aus FOSS bestehen, als Ganzes den Bedingungen der FOSS mit Copyleft Effekt unterliegen. Spätere Rechtsprechung hat diesen Fehler erkannt und auch unter dem Gesichtspunkt der Verwirkung und dem Unclean-Hands-Einwand Einwendungen gegen das Verwertungsrecht zurückgewiesen.[7]

29 Die Möglichkeiten für Lizenznehmer, wegen FOSS Verstößen Kostenfreiheit zu erlangen, sind etwas kleiner als angenommen. FOSS Lizenzen können nur die Instrumente des Urheberrechts einsetzen. Das Sanktionsarsenal beschränkt sich daher darauf, das kostenlos gewährte Nutzungsrecht an der FOSS entfallen zu lassen. Die Lizenz kann keine zusätzlichen Nutzungsrechte für jedermann an proprietärer Software erzeugen.

bb) Ungenutztes Schädigungspotenzial der Rechtsinhaber

30 Die „offene" Verbreitung von FOSS bietet Vorteile, aber zugleich nicht zu unterschätzende Risiken, weniger für den Anwender als vielmehr für denjenigen, der FOSS für die eigene Entwicklung einsetzen und auch an Endkunden vertreiben will.

(1) Rechtlicher Fokus

31 Die Lizenzbedingungen der jeweiligen FOSS legen die urheberrechtlichen Rahmenbedingungen fest, unter denen der Entwickler der Software bzw. deren Rechtsinhaber anderen Nutzungsrechte an der FOSS einräumen will. Da die FOSS Lizenzbedingungen auf diese Weise die

6 LG Berlin, 8.11.2011 – 16 O 255/10, ZUM-RD 2012, 153.

7 OLG Karlsruhe, 27.1.2021 – 6 U 60/20, K&R 2021, 271 ff.; LG Mannheim, 24.1.2020 – 7 O 71/19 (abrufbar unter: https://shop.ruw.de/download-FOSS). Hierzu ausführlich Rn. 612 ff.

Ausübung des Urheberrechts konkretisieren, stellt jede Verletzung der Lizenzbedingungen eine Urheberrechtsverletzung dar, die urheberrechtliche Ansprüche der §§ 69f, 69a Abs. 4 i. V. m. §§ 97 ff. UrhG auslöst, sofern die jeweiligen Anspruchsvoraussetzungen gegeben sind (siehe auch Rn. 499 ff.).[8]

Automatischer Rechtsfortfall oder inhaltliche Beschränkung? **32**

Die Geltendmachung von urheberrechtlichen Ansprüchen durch den Urheber wegen lizenzwidriger FOSS Nutzung hängt im Wesentlichen davon ab, ob das lizenzwidrige Verhalten zu einem Wegfall der durch die FOSS Lizenz eingeräumten Nutzungsrechte führt ober ob eine bloße Verletzung vertraglicher Pflichten vorliegt, was den Rechtsinhaber auf die bloße Durchsetzung von schuldrechtrechtlichen Ansprüchen gegen den Verletzer beschränken würde. Beispielsweise regelt die GPL-2.0 in Ziff. 4 einen automatischen Rechtsfortfall bei lizenzwidriger Nutzung, stellt aber lediglich eine vertragliche Regelung dar. Daher wird in der Literatur diskutiert, welche rechtlichen Folgen sich aus dieser Klausel ergeben.

Während einige Autoren von einer inhaltlichen Beschränkung der Nutzungsrechte i. S. d. § 31 Abs. 1 Satz 2 UrhG ausgehen,[9] nimmt die h. M. eine auflösend bedingte Rechtseinräumung nach § 158 Abs. 2 BGB an.[10] Im Ergebnis kann der GPL-2.0 nicht nur eine schuldrechtliche Verpflichtung zur Einhaltung der Lizenzbedingungen entnommen werden, so dass die h. M. zutreffend davon ausgeht, dass dem Lizenznehmer die Nutzungsrechte unter der auflösenden Bedingungen eingeräumt werden, dass dieser sich an die Lizenzbedingungen hält (siehe Rn. 152 f.). Wird gegen die Pflichten verstoßen, fallen die Nutzungsrechte automatisch ex nunc weg und der Lizenznehmer begeht eine Urheberrechtsverletzung, wenn er die Software entgegen den Lizenzbedingungen vertreibt.[11]

---

8 Kurzüberblick bei *Van den Brande/Coughlan/Jaeger*, S. 137 f.; weiterführend *Jaeger/Metzger*, Open Source Software, Rn. 209 ff. m. w. N.

9 Siehe Nachweise bei *Grützmacher*, in: Wandtke/Bullinger, UrhG, § 69c Rn. 116.

10 Weiterführend *Teupen*, S. 207 ff.; *Jaeger*, in: ifrOSS, Die GPL kommentiert und erklärt, Ziff. 4 Rn. 11 ff.; *Jaeger/Metzger*, Open Source Software, Rn. 209; *Grützmacher*, in: Wandtke/Bullinger, UrhG, § 69c Rn. 116, jeweils mit m. w. N.

11 *Jaeger/Metzger*, Open Source Software, Rn. 211; *Metzger/Jaeger*, GRUR Int. 1999, 839 843.

(2) Realitätsfokus

33 Eine „Massenabmahnwelle" der FOSS Entwickler ist bisher nicht losgebrochen, aber in Anbetracht des doch weitreichenden FOSS Einsatzes in Produkten aller Art künftig nicht per se ausgeschlossen. Die Konsequenzen aus der auch gerichtlichen Verfolgung der bei lizenzwidrigem FOSS Einsatz entstehenden Ansprüche können recht weitreichend sein. So ist u. U. der Rückruf ganzer Produktserien, in denen FOSS Komponenten ohne Einhaltung der Lizenzbedingungen implementiert sind, nicht undenkbar. Dies kann nicht zuletzt immense wirtschaftliche Schäden und Reputationsverluste für die in Anspruch genommenen Unternehmen nach sich ziehen.

34 Die entstehenden rechtlichen Risiken sind abhängig von der Strenge der jeweiligen FOSS Lizenz und von dem Pflichtenprogramm, das von den Lizenzbedingungen vorgeschriebenen ist und durch eine Weitergabe der jeweiligen FOSS z. B. in einem Endprodukt ausgelöst wird. Dies betrifft v. a. den Vertrieb der FOSS an Endkunden. Das rechtliche Risiko wird zusätzlich erhöht, wenn FOSS Komponenten mit proprietären Programmkomponenten z. B. verbunden im Rahmen von sogenannten Value added Produkten[12] vertrieben werden. Handelt es sich bei der verbundenen oder auch nur bei Installation und Ausführung interagierenden FOSS um eine solche mit angeordnetem strengem Copyleft (wie z. B. bei der GPL-2.0 oder GPL-3.0 der Fall), so ist u. U. der Vertrieb nur gestattet, wenn alle Programme, die von der GPL Software abgeleitet sind, ebenfalls unter die GPL gestellt werden (siehe weiterführend Rn. 234 ff.).

35 Die deutsche Rechtsprechung hat noch immer keinen gemeinsamen Nenner gefunden; v. a. zur Reichweite des Copyleft Effekts der GPL fehlen Entscheidungen, soweit ersichtlich, bisher völlig.[13] Jedoch ist

---

12 Zum Begriff *Jaeger/Metzger*, Open Source Software, Rn. 22 sowie Rn. 51; *Wiebe*, in: Spindler/Schuster, § 69c UrhG Rn. 55.

13 Das OLG Hamburg reißt in dem Rechtsstreit Hellwig ./. VMWare (OLG Hamburg, 28.2.2019 – 5 U 146/16, MMR 2019, 452 Rn. 76 m. Anm. *Galetzka/Hackel*) die Problematik zumindest an, ohne dass sie sich jedoch entscheidungserheblich auswirkt. Mit der gleichen Feststellung die Vorinstanz (LG Hamburg, MMR 2016, 740, Rn. 95 f. m. Anm. *Galetzka/Otto*). Auch LG Mannheim, 24.1.2020 – 7 O 71/19 (abrufbar unter: https://shop.ruw.de/download-FOSS) und OLG Karlsruhe, 27.1.2021 – 6 U 60/20, K&R 2021, 271 ff. mussten die Problematik nicht entscheiden, siehe hierzu. Rn. 612 ff.

ein Großteil der Entscheidungen der letzten Jahre (siehe Rn. 36 ff. und ausführlich Anhang Rn. 837.)[14] zugunsten der Rechtsinhaber von FOSS ergangen, die von den jeweiligen Prozess-Kontrahenten ohne Beachtung der jeweiligen FOSS Lizenzbedingungen und damit urheberrechtswidrig eingesetzt worden ist. Die bisherigen FOSS Urteile schärfen daher den Fokus hinsichtlich der sich tatsächlich auch in der Praxis realisierenden Rechtsrisiken und fördern immer mehr das Bedürfnis von Unternehmen unterschiedlichster Wirtschaftszweige nach FOSS Compliance. Diese verfolgt das Ziel, einen kontrollierten Einsatz von FOSS und deren rechtskonformen Vertrieb v. a. in Verbindung mit Value added Produkten oder in eingebetteten Systemen zu ermöglichen.

d) Was bisher geschah…

In den letzten Jahren sind einige Vertreter aus der Open Source Community (siehe auch Rn. 48 ff.) aufgetreten, die das Bild ergangener Gerichtsentscheidungen bzgl. der Verfolgung von Rechtsverstößen im FOSS Bereich geprägt haben. Außerdem wurden einige interessante Verfahren in Übersee im FOSS Bereich ausgetragen. **36**

aa) Harald Welte: Pionier der deutschen FOSS Rechtsprechung

Allen voran und gleichsam als Initiator einiger richtungsweisender FOSS Urteile in Deutschland zu nennen ist *Harald Welte*, der Miturheber und -entwickler des Linux kernel. *Welte* betreibt unter der URL gpl-violations.org eine zentrale Informations- und Community-Webseite zu FOSS und katalogisiert dort zahlreiche Fälle, in denen die GPL (insbesondere die GPL-2.0) in der Praxis verletzt wurde. Ein Überblick der von ihm auch gerichtlich verfolgten FOSS Verletzungsfälle und weitere FOSS News findet sich ebenfalls auf der genannten Community-Webseite.[15] **37**

Die Klärung von einigen wesentlichen Grundsatzfragen im FOSS Bereich für das deutsche (Urheber-)Recht vor den deutschen Gerichten geht auf von *Welte* angestrengte gerichtliche Verfahren zurück. **38**

---

14 Instruktiver Überblick bei *Van den Brande/Coughlan/Jaeger*, S. 142 ff.
15 https://gpl-violations.org/news/.