SSOAR

Open Access Repository
www.ssoar.info

Open Source Softwareprojekte zwischen Passion und Kalkül

Schrape, Jan-Felix

Veröffentlichungsversion / Published Version
Arbeitspapier / working paper

Empfohlene Zitierung / Suggested Citation:
Schrape, J.-F. (2015). *Open Source Softwareprojekte zwischen Passion und Kalkül.* (Research contributions to organizational sociology and innovation studies / Stuttgarter Beiträge zur Organisations- und Innovationssoziologie : SOI discussion paper, 2015-02). Stuttgart: Universität Stuttgart, Fak. 10 Wirtschafts- und Sozialwissenschaften, Institut für Sozialwissenschaften Abt. VI Organisations- und Innovationssoziologie. https://nbn-resolving.org/urn:nbn:de:0168-ssoar-453454

Nutzungsbedingungen:
Dieser Text wird unter einer CC BY Lizenz (Namensnennung) zur Verfügung gestellt. Nähere Auskünfte zu den CC-Lizenzen finden Sie hier:
https://creativecommons.org/licenses/by/4.0/deed.de

Terms of use:
This document is made available under a CC BY Licence (Attribution). For more Information see:
https://creativecommons.org/licenses/by/4.0

gesis Leibniz-Institut für Sozialwissenschaften

Mitglied der Leibniz-Gemeinschaft

Stuttgarter Beiträge zur Organisations- und Innovationsforschung

SOI Discussion Paper 2015-02

Open Source Softwareprojekte zwischen Passion und Kalkül

Jan-Felix Schrape

Universität Stuttgart
Institut für Sozialwissenschaften
Organisations- und Innovationssoziologie

Jan-Felix Schrape
Open Source Softwareprojekte zwischen Passion und Kalkül
SOI Discussion Paper 2015-02

Universität Stuttgart
Institut für Sozialwissenschaften
Abteilung für Organisations- und Innovationssoziologie (SOWI VI)
Seidenstr. 36
D-70174 Stuttgart
http://www.uni-stuttgart.de/soz/oi/

Herausgeber

Prof. Dr. Ulrich Dolata
Tel.: 0711 / 685-81001
ulrich.dolata@sowi.uni-stuttgart.de

Redaktion

Dr. Jan-Felix Schrape
Tel.: 0711 / 685-81004
felix.schrape@sowi.uni-stuttgart.de

Stuttgarter Beiträge zur Organisations- und Innovationsforschung (SOI)
Discussion Paper 2015-02 (11/2015)
ISSN 2191-4990

© 2015 by the author(s)

Jan-Felix Schrape ist wissenschaftlicher Mitarbeiter der Abteilung Innovations- und Organisationssoziologie am Institut für Sozialwissenschaften der Universität Stuttgart.

felix.schrape@sowi.uni-stuttgart.de

Weitere Downloads der Abteilung für Organisations- und Innovationssoziologie am Institut für Sozialwissenschaften der Universität Stuttgart finden sich unter:

http://www.uni-stuttgart.de/soz/oi/publikationen/

Zusammenfassung

Dieses Papier entwickelt auf der Grundlage von aggregierten Marktdaten, Dokumentenanalysen sowie Literaturauswertungen einen systematisierenden Überblick über Open Source Software Communities und ihre sozioökonomischen Kontexte. Nach einer Rekonstruktion der Herausbildung quelloffener Entwicklungsvorhaben werden die Beziehungen zwischen Open Source Projekten und etablierten Technologiekonzernen diskutiert. Daran anknüpfend werden entlang ihrer Koordinationsweisen und dem Grad ihrer Unternehmensnähe vier typische Varianten derzeitiger Open Source Gemeinschaften voneinander abgegrenzt. Insgesamt zeigt sich, dass die quelloffene Softwareentwicklung inzwischen zu einer allgemeinen Branchenmethode avanciert ist, dabei aber ihre Formatierung als Gegenentwurf zur kommerziellen und proprietären Herstellung weitgehend verloren hat. Während freie Software zunächst subversiv konnotiert war, ist das Involvement in Open Source Projekte heute zu einem festen Bestandteil der Innovationsstrategien aller großen Softwareanbieter geworden.

Abstract

Based on a review and evaluation of business reports, market data, literature, documents and press releases, this paper aims to provide a systematic overview of open source communities, their predominant coordination modes and their socio-economic contexts. The investigations highlight the extend to which successful free and open source software development projects today are dependent on the involvement of commercial companies. Open Source projects are by now deeply entrenched in the technology sector; they have been stripped of their subversive connotations and have become part of the innovation strategies of all established software providers.

Inhalt

1 Einleitung 5

2 Historische Entwicklung 7

2.1 Kollektive Invention (1950er Jahre) 8
2.2 Kommodifizierung – Subkulturgenese (1960/70er Jahre) 9
2.3 Institutionalisierung (1980/90er Jahre) 12
2.4 Wachstum – Diversifizierung (2000er Jahre) 16

3 Open Source und kommerzieller Markt 20

3.1 Marktrelevanz quelloffener Software 21
3.2 Involvement etablierter Anbieter 24
3.3 Open Source Companies 27
3.4 Patronage und Spendenfinanzierung 29

4 Spielarten quelloffener Softwareprojekte 32

4.1 Korporativ geführte Kollaborationsprojekte 34
4.2 Heterarchisch angelegte Infrastrukturvorhaben 37
4.3 Elitezentrierte Projektgemeinschaften 40
4.4 Peer Production Communities 43

5 Bilanz: Open Source als Utopie, Methode und Innovationsstrategie 47

Literatur 51

1 Einleitung

Quelloffene Softwarekomponenten spielen für die basalen Infrastrukturen des Internets eine zentrale Rolle: Der Apache HTTP Server ist seit 1996 die meistverbreitete Webserver-Software; linuxbasierte Architekturen dominieren den Markt für Server-Betriebsumgebungen; der Großteil der im Netz eingesetzten Content-Management- und Datenbanksysteme steht unter freien Lizenzen (Netcraft 2015; W3techs 2015). Das Marktforschungsunternehmen Gartner beobachtet eine vermehrte Nutzung von Open Source Software in Organisationen und geht davon aus, dass 2016 die Mehrheit aller IT-affinen Firmen auch in kritischen Bereichen auf quelloffene Elemente zurückgreifen wird (Driver 2014). Und auf dem Feld der Consumer Software sind Produkte, die vollständig oder teilweise auf Open Source Projekten basieren, zu einem festen Bestandteil des Alltags vieler Anwender geworden, so etwa der Browser Mozilla Firefox oder das mobile Betriebssystem Android. Auch wenn sich eines der vorrangigen Ziele des ‚free software movements' – die Ablösung von Microsoft Windows durch GNU/Linux auf dem Desktop – gemessen an den globalen Verbreitungszahlen für Mitte 2015 (Windows: 91 Prozent; Mac OS X: 7 Prozent; Linux: 1,6 Prozent) bislang nicht erfüllt hat (NetApplications 2015), kann quelloffener Software heute in vielen Segmenten des IT-Marktes eine hohe Relevanz zugeschrieben werden.

Diese zunehmende Bedeutung von Open Source Projekten in der Softwareentwicklung wurde in den Sozialwissenschaften angesichts klassischer Sichtweisen, die *intellectual property rights* als wesentliche Treiber in Innovationsprozessen ansehen (Arrow 1962; Romer 1990), zunächst mit Erstaunen zur Kenntnis genommen (Lessig 1999) und nachfolgend von einigen Autoren als Beleg für die Emergenz eines neuen Produktionsmodells formatiert, das auf freiwilliger wie selbstgesteuerter Kollaboration beruht, den Stellenwert korporativer Akteure in der Arbeitswelt schmälert und eingespielten Formen ökonomischer Koordination auf Dauer überlegen sein könnte (Lakhani/Hippel 2003; Osterloh/Rota 2007; Suddaby 2013). Insbesondere die durch Yochai Benkler popularisierte Vorstellung der ‚commons-based peer production' als technisch effektivierte „collaboration among large groups of individuals [...] without relying on either market pricing or managerial hierarchies to coordinate their common enterprise" (Benkler/Nissenbaum 2006: 394), die mit „systematic advantages [...] in identifying and allocating human capital/creativity" einhergehen soll (Benkler 2002: 381), erfuhr eine intensive Reflexion und Anwendung auf angrenzende Kontexte (z.B. Kostakis/Papachristou 2014; Rifkin 2014; Baldwin/Hippel 2011).

Zweifelsohne lassen sich unter den inzwischen millionenfach initiierten Open Source Projekten (BlackDuck/NorthBridge 2015) Entwicklergemeinschaften finden, die sich durch mehr oder minder ausgeprägte Muster einer „radically decentralized, collaborative, and nonproprietary [...] peer production" (Benkler 2006: 60) auszeichnen und deren Teilhabende sich primär aus intrinsischen Motiven einbringen – „satisfying psy-

chological needs, pleasure, and a sense of social belonging“ (Benkler 2004: 1110f.). Empirische Untersuchungen legen allerdings nahe, dass die meisten marktrelevanten Open Source Projekte mittlerweile in erster Linie durch die Beiträge von angestellten Softwareentwicklern getragen werden. In der oft als typisches Beispiel herangezogenen Linux Kernel Community etwa wurden laut einer Erhebung der Linux Foundation (Corbet et al. 2015: 11) zuletzt über 80 Prozent der Entwicklung von Programmierern durchgeführt, „who are being paid for their work“ – unter anderem von Google, IBM und Intel, die mit Blick auf ihre Geschäftsaktivitäten an der Weiterentwicklung des Kernels interessiert sind. Darüber hinaus stehen viele der dauerhaft aktiven Open Source Projekte in einem Verweisungszusammenhang mit kommerziellen Angeboten oder werden über Spenden an ihre Dachstiftungen durch Unternehmen mitfinanziert.

In Anbetracht dieser Verschränkungen von quelloffener und proprietärer Softwareentwicklung, deren unternehmerische Vor- und Nachteile in der Managementforschung unter Stichworten wie „open innovation“ (West et al. 2014) diskutiert werden, reichen die in den Sozialwissenschaften oft üblichen eher pauschalen Verweise auf Open Source Communities als Alternative oder als Gegenentwurf zur kommerziellen Softwareentwicklung nicht mehr aus, um dem breiten Spektrum an sehr unterschiedlich ausgerichteten Projekten in diesem Bereich gerecht zu werden. Dies zeigt sich nicht zuletzt in den Deutungskonflikten, welche sich entlang von Begriffen wie ‚Free Software‘, ‚Open Source Software‘ und ‚Free/Libre Open Source Software‘ entsponnen haben, sowie in den Kontroversen zwischen den zwei großen Bezugsorganisationen für quelloffene Software – der pragmatisch orientierten Open Source Initiative und der gesellschaftspolitisch fundierten Free Software Foundation.

Das vorliegende Papier verfolgt insofern das Ziel, auf der Grundlage von aggregierten Marktdaten, Dokumentenanalysen, Literaturauswertungen sowie Hintergrundgesprächen mit Softwareentwicklern einen systematisierenden Überblick über Open Source Communities und ihre sozioökonomischen Kontexte zu entfalten. Zunächst erfolgt eine Rekonstruktion der Herausbildung quelloffener Projekte, die aufzeigt, dass freie und proprietäre Softwareentwicklung seit jeher ineinandergreifen (*Kap. 2*). Nachfolgend werden die Beziehungen zwischen Open Source Communities und etablierten IT-Unternehmen diskutiert (*Kap. 3*). Daran anknüpfend werden vier typische Varianten derzeitiger Open Source Projekte voneinander abgegrenzt – von korporativ geführten Kollaborationsprojekten und elitezentrierten Projektgemeinschaften über heterarchisch angelegte Infrastrukturvorhaben bis hin zu egalitär ausgerichteten Entwicklergruppen, die am ehesten der Idee einer ‚commons-based peer production‘ entsprechen (*Kap. 4*). Abschließend werden die sich wandelnden Relationen zwischen freier Software und allgemeinem Markt in verdichteter Form nachgezeichnet: Während die quelloffene Softwareentwicklung zunächst subversiv konnotiert war und in geschützten Nischen stattfand, ist das Involvement in Open Source Projekte heute zu einem festen Bestandteil der Innovationsstrategien aller großen Anbieter geworden (*Kap. 5*).

2 Historische Entwicklung

Dass Unternehmen ihre Wissensbestände insbesondere zu Beginn von Innovationsprozessen auch außerhalb formaler Kooperationsbeziehungen miteinander teilen, ist kein neuartiges Phänomen und wurde bereits durch Robert C. Allen (1983: 21) als weithin unerforschter Prozess der *collective invention* beschrieben: „To the degree that economists have considered this behaviour at all, it has been regarded as an undesired ‚leakage' that reduces the incentives to invent. That firms desire such behaviour and that it increases the rate of invention [...] are possibilities not yet explored." Von den von Robert C. Allen und Kollegen diskutierten Beispielen (Tab. 1) hebt sich das Open Source Modell allerdings durch zwei entscheidende Merkmale ab (Cowan/Jonard 2003; Osterloh/Rota 2007): Zum einen unterliegt der Austausch in onlinebasierten Arbeitsgemeinschaften keiner geographischen Begrenzung mehr und bleibt nicht auf spezifische Branchenkontexte beschränkt. Zum anderen haben lizenzrechtliche Konstruktionen wie die GNU General Public License, die absichert, dass auch Derivate freier Software ausschließlich unter den gleichen Bedingungen weitergegeben werden können (‚Copyleft'), sowie die im Web erleichterte Durchsetzung informeller Regeln und Konventionen dazu beigetragen, dass Open Source Projekte im Gegensatz zu früheren Ausprägungen kollektiver Invention auch nach der Herausbildung eines dominanten Designs und dessen kommerzieller Verwertung überlebensfähig bleiben.

Tabelle 1: Historische Beispiele für ‚collective invention'

Episode	Austauschprozesse	Resultat
The Cornish Pumping Engine ca. 1810–1850, Cornwall/London, England (Nuvolari 2004)	Austausch technischer Daten; Vergleich der individuellen Fortschritte über Fachjournale	Entwicklung einer brennstoffsparenden Hochdruck-Dampfmaschine für den Bergbau
Papierherstellung ca. 1827–1857, New England, USA (McGaw 1987)	Stabile Gemeinschaft von Mühlenbesitzern; regelmäßiger informeller Erfahrungsaustauch	Erhöhung der Produktivität durch zunehmende Mechanisierung der Produktion
Hochofentechnologie ca. 1850–1880, Cleveland District, England (Allen 1983)	Proaktiver Wissensaustausch über Fachpublikationen/-gesellschaften; kollektiver Trial- & Error-Prozess	Verringerung der Energiezufuhr durch Steigerung der Bauhöhen und Temperaturanpassungen
Flachbildschirme (LCD, Plasma) ca. 1969–1989, Japan/Europa/ Nordamerika (Spencer 2003)	Wissenschaftliche Publikation von Ergebnissen aus firmeneigener Forschung und Entwicklung	Inkrementelle Verbesserung und Technologieentwicklung in der vorkommerziellen Phase
Homebrew Computer Club ca. 1975–1978 [1986], Silicon Valley, USA (Meyer 2003)	Austausch technischer Informationen bis zu den ersten Markterfolgen ausgegründeter Firmen	Entwicklung erster marktfähiger persönlicher Mikrocomputer und anderer IT-Produkte

Vor diesem Hintergrund lässt sich die Geschichte der quelloffenen Softwareentwicklung in vier Stadien einteilen: (1) In den 1950er-Jahren, als Software noch nicht als separates Produkt galt, sondern zusammen mit der Hardware vertrieben wurde, folgte die Zusammenarbeit von universitären und korporativen Forschern freien akademi-

schen Prinzipien (*kollektive Invention*). (2) Ab Mitte der 1960er Jahre bildete sich eine eigenständige Softwareindustrie heraus und der Anteil restriktiv lizenzierter Software stieg an. Gleichzeitig entstanden frühe Interessengruppen von Computerhobbyisten (*Kommodifizierung – Subkulturgenese*). (3) Die elektronische Vernetzung förderte ab 1980 die ortsunabhängige Verdichtung freier Entwicklergemeinschaften, die ab 1985 durch die Free Software Foundation unterstützt wurden. Daneben wurden erste ‚Copyleft'-Lizenzen spezifiziert (*Institutionalisierung*). (4) Um freie Software von ihren ideologischen Konnotationen zu lösen, schufen Szeneprotagonisten ab 1998 das Label ‚Open Source' und gründeten die Open Source Initiative. Die durch sie zertifizierten flexibleren Lizenzen trugen dazu bei, die sozioökonomische Relevanz quelloffener Software weiter zu steigern (*Wachstum – Diversifizierung*).

2.1 Kollektive Invention (1950er Jahre)

Die ersten in Serie hergestellten digitalen Computer wurden in enger Kooperation zwischen staatlichen bzw. universitären Organisationen und privatwirtschaftlichen Unternehmen entwickelt, die ihr Geld in erster Linie mit der Produktion der kostspieligen Hardware verdienten. Der ab 1951 vorrangig in administrativen Einrichtungen installierte Großrechner UNIVAC I etwa schlug mit bis zu 1,5 Mio. US-Dollar zu Buche; die ab 1953 rund zweitausendmal verkaufte IBM 650 Magnetic Drum Data-Processing Machine kostete 1959 noch ca. 150.000 US-Dollar (IBM 1959; Ceruzzi 1998). Frühe Computerprogramme wurden mit Source Code (den für den Menschen lesbaren Text eines Programms) weitergegeben, um Kunden die Möglichkeit zu bieten, die teuren Mainframe-Systeme an ihre Bedürfnisse anzupassen. ‚Software' wurde allenfalls als weitgefasstes Antonym zu Hardware gebraucht – beispielsweise um nichttechnische Fehlerquellen wie „human factors in reliability" zu umschreiben (Carhart 1953: 69). Erstmals im heutigen Sinne verwendet wurde der Ausdruck von John Tukey (1958: 2): „Today the ‚software' comprising the carefully planned interpretive routines, compilers, and other aspects of automative programming are at least as important to the modern electronic calculator as its ‚hardware' [...]."

Dass Software in dieser Anfangszeit der IT-Branche kein eigenständiger Marktwert zugesprochen wurde, liegt auch darin begründet, dass die einzelnen Computersysteme nicht kompatibel und deren situationsspezifische Einsatzpotentiale noch kaum erforscht waren, was nach einer intensiven Kooperation zwischen Anwendern und anbieterseitigen Spezialisten verlangte, um das jeweilige System überhaupt nutzbar zu machen (Phister 1976). Dieser hohe individuelle Programmieraufwand wurde von der International Business Machines Corporation (IBM) nach ihrem Einstieg in den Markt für digitale Computer 1952 als gravierendes Hemmnis für den Ausbau ihres Hardware-Geschäfts eingestuft, weshalb IBM mit SHARE bereits Mitte der 1950er Jahre eine Gesellschaft für den Austausch von technischen Informationen zwischen Nutzern von IBM-Systemen mitbegründete. Ihre anfänglich zwei Dutzend Mitglieder

entwickelten gemeinsam Modifikationen und Erweiterungen, allerdings wuchsen die Transaktionskosten in Relation zu den Produktivitätsgewinnen mit steigender Beteiligtenzahl disproportional an, was mit ein Grund dafür war, dass das SHARE Modell nicht zu einem Branchenstandard avancierte (Campbell-Kelly 2003; Brooks 1975).

Mit Unterstützung der SHARE Gruppe entwickelte IBM die abstrakte Programmiersprache FORTRAN, die via Compilern erstmals die Möglichkeit bot, das gleiche Programm auf unterschiedlichen Systemen auszuführen, und gab den Programmcode mit Dokumentation ab 1957 kostenfrei an Kunden wie Mitbewerber ab, um – ähnlich wie in Beta-Programmen heute – praxisnahe Optimierungen zu ermöglichen und die eigene Marktposition zu stärken (Bashe et al. 1986). Systemübergreifende Sprachen wie FORTRAN senkten die Kosten in der Programmierung und machten mit ersten Vereinheitlichungen im Hardwarebereich die Entwicklung standardisierter Applikationen durch Drittanbieter denkbar. Die erste kommerziell vertriebene Software war das Dokumentationssystem AUTOFLOW der Firma Applied Data Research zur Generierung von Programmablaufplänen, das ab 1965 für 2.400 US-Dollar angeboten wurde und 1968 das erste Patent für ein Computerprogramm erhielt. Viele IBM-Kunden konnten mit dem Konzept einer separat verkauften Anwendung jedoch zunächst wenig anfangen: „Over and over again, as a result of a sales presentation on AUTOFLOW, the potential customer would call his IBM account representative and ask when they were going to upgrade their program to do what AUTOFLOW did.“ (Johnson 2002: 107)

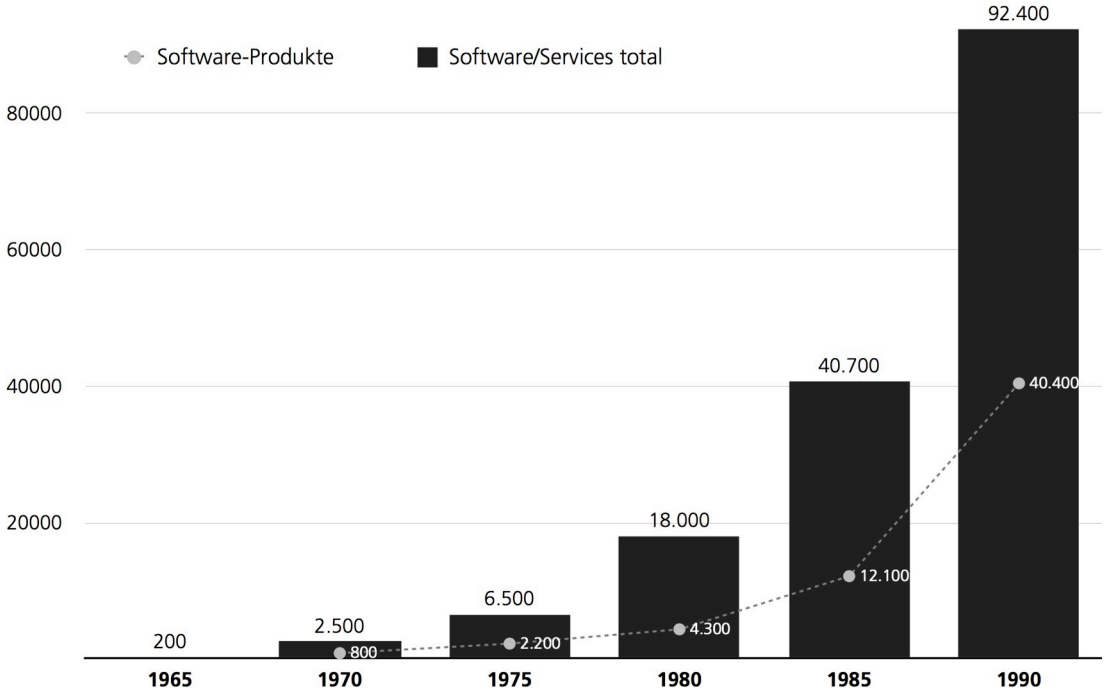
Bis in die 1960er Jahre hinein wurde Software weder von Anbietern noch von Kunden als ein von der Hardware unabhängiges Produkt wahrgenommen, sondern „as a research tool to be developed and improved by all users“ (Gulley/Lakhani 2010: 6). Dementsprechend lässt sich die frühe Softwareentwicklung als Spielart der durch Allen (1983) beschriebenen *collective invention* fassen: als eher informell strukturierte Kollaboration zwischen Produzenten und staatlichen bzw. universitären Einrichtungen, deren Kreis ob der erforderlichen Ressourcen und Wissensbestände zunächst begrenzt blieb. Bereits in dieser Zeit nahm das Unternehmen IBM indes eine herausgehobene Stellung ein: 1960 erwirtschaftete es Einnahmen von 1,8 Mrd. US-Dollar und produzierte 65 Prozent aller Computer auf dem US-amerikanischen Markt – ein Anteil, der sich bis 1965 auf knapp 75 Prozent steigern sollte (Mulligan 2013: 107).

2.2 Kommodifizierung – Subkulturgenese (1960/70er Jahre)

Eine kommerzielle Softwareindustrie entwickelte sich zunächst primär in den USA (Mowery 1999). Da weder das Urheberrecht noch das Patentwesen auf die neue Produktkategorie eingestellt waren, blieb der Handel mit Software in den ersten Jahren freilich ein risikoreiches Geschäft, bis Ende der 1960er Jahre erste Gerichtsurteile Rechtssicherheit schafften. Auftrieb erhielt die junge Branche nicht zuletzt durch ein 1969 eingeleitetes kartellrechtliches Verfahren, in welchem IBM vorgeworfen wurde,

durch das kombinierte Angebot von Hardware, Software und Dienstleistungen potentielle Mitbewerber aus dem Rennen werfen zu wollen. IBM kündigte daraufhin an, seine Angebote zu entbündeln, was sich zunächst weder lizenzarchitektonisch unkompliziert umsetzen noch unfallfrei an die Kunden vermitteln ließ (Burton 2002). Durch dieses und ähnliche Verfahren wurde Software zunehmend als eigenständiges Produkt sichtbar und rechtlich einfasst. In den nachfolgenden Jahren erlebte die US-amerikanische Softwareindustrie ein rasantes Wachstum (Abb. 1).

Abbildung 1: Umsätze der US-amerikanischen Softwarebranche (Mio. US-Dollar)



Datenquelle: Schätzungen der Computer & Business Equipment Manufacturers Association, Basis-USD-Werte für 1990 (Sayadian 1990; Juliussen/Juliussen 1990).

Während 1966 rund 50 unabhängige Unternehmen in der US-Softwarebranche tätig waren, stieg deren Zahl bis 1969 auf über 2.800 an, darunter als erfolgreichster Anbieter die Computer Sciences Corporation, die ihren Umsatz unter anderem mit Finanzverwaltungssoftware von 6 Mio. US-Dollar im Jahr 1964 auf 82 Mio. US-Dollar im Jahr 1970 steigern konnte (Fisher et al. 1983). Der Löwenanteil der Erlöse wurde aber schon damals in der In-House-Entwicklung und im Dienstleistungsgeschäft gemacht. Dass sich überhaupt ein nennenswerter Markt für Standardsoftware entwickeln konnte, lässt sich neben der durch IBM beförderten Vereinheitlichung der Mainframe-Systeme auf das Entstehen einer neuen Produktkategorie zurückführen: den schrankgroßen *minicomputer*. Das erste verkaufte Gerät dieses Typs war ab 1960 der DEC PDP-1, der 25.000 Operationen pro Sekunde ausführen konnte (0,025 MHz) und sich im Gegensatz zu früheren Systemen durch eine Einzelperson steuern ließ. Mitte

der 1960er Jahre waren Minicomputer für unter 20.000 US-Dollar zu haben; ihre Absatzzahlen überstiegen die der Mainframes Ende des Jahrzehnts (Steinmueller 1995).

Für die weitere Entwicklung der Softwarebranche und die Ausbildung einer computerzentrierten Subkultur spielte die Verbreitung von Minicomputern eine wichtige Rolle: Zum einen waren sie im Betrieb deutlich günstiger als Mainframes, daher nicht auf eine möglichst effiziente Nutzung ausgelegt und leichter zugänglich; zum anderen ermöglichten der PDP-1 und seine Nachfolger die Verwendung neuer Ein- und Ausgabeschnittstellen, welche die Interaktion mit dem Nutzer vereinfachten und die Herausbildung neuer Softwaregenres (z.B. Grafikverarbeitung) beförderten. So wurden Fernschreiber für die Texteingabe und den Druck eingesetzt und Kathodenstrahlröhrenmonitore angeboten, die sich via *lightpen* bedienen ließen. Eines der bekanntesten PDP-1-Programme ist das Spiel ‚Spacewar', das ab 1961 durch Mitglieder des studentischen Tech Model Railroad Clubs am MIT für den von DEC an das Institut gespendeten Minicomputer entwickelt wurde und aufgrund seiner Demonstrationspotentiale bald zur Standardsoftware des PDP-1 gehörte (Lowood 2009).

Vor allen Dingen im nordamerikanischen akademischen Milieu, in dem sich *computer science* als Disziplin bereits Ende der 1950er Jahre etabliert hatte, während die Informatik in Deutschland erst 10 Jahre später entstand, boten stetig günstigere und institutsöffentlich erfahrbare Minicomputer einen idealen Nährboden für informelle Projektgruppen, deren Teilhabende sich *hackers* nannten und die Limitationen vorhandener Computersysteme zu überwinden suchten (Levy 1984). Auch die bis in die 1990er Jahre populäre Programmiersprache BASIC wurde ab 1964 im Hochschulkontext entwickelt und kostenfrei abgegeben. Da BASIC rasch zu erlernen war, bildete es mit den ersten Mikrocomputern wie dem ab 1975 zehntausendfach verkauften Altair 8800 (ab 400 US-Dollar) eine Grundlage für die sich in den 1970er Jahren entlang populärwissenschaftlicher Zeitschriften herausbildende Amateur Computing Szene: „It is these basement Edisons [...] who will most probably realize the vast potential of the silicon chip for the consumer [...]. They are [...] anxious to share technological insights and applications with other chip fanatics." (Time 2/1978: 49)

Ein Problem, das für kommerzielle Softwareanbieter mit diesem neuen Markt einherging, bestand darin, dass Programme durch Computerhobbyisten zwar gerne weitergegeben, aber nur selten käuflich erworben wurden. Darüber beschwerte sich der junge Software-Entrepreneur Bill Gates (1976) mit Blick auf das durch sein Unternehmen Micro-Soft verkaufte Altair BASIC in einen offenen Brief wie folgt: „As the majority of hobbyists must be aware, most of you steal your software. Hardware must be paid for, but software is something to share. Who cares if the people who worked on it get paid?" Gates' Vorwürfe provozierten eine Vielzahl an Reaktionen, die argumentierten, (1) dass Mikrocomputer-Käufer ein funktionierendes System ohne Zusatzkosten erwarten könnten; (2) dass Altair BASIC auf Universitätsrechnern erstellt wurde und auf frei verfügbaren Vorarbeiten basiere; (3) oder dass es mithin an den Nutzern

selbst sei, kostenfreie Software zu entwickeln (Singer 1976; Warren 1976). Ein Beispiel für ein solches nutzerzentriertes Projekt aus dieser Zeit ist Tiny BASIC, welches durch Mitglieder der Standford University initiiert wurde und von einigen Autoren als Vorbote des ‚free software movements' angesehen wird (Driscoll 2015).

Ebenfalls eng mit dem universitären Milieu verknüpft war die Entwicklung des Betriebssystems Unix, das ab 1969 durch die AT&T Bell Labs entworfen wurde (Holtgrewe/Werle 2001; Weber 2005): 1973 öffentlich vorgestellt, stieß Unix auf ein breites Interesse, denn da es in der Programmiersprache C verfasst war, konnte es auf nahezu allen Computern eingesetzt werden, wodurch Systeme unterschiedlicher Anbieter Kompatibilität erlangten und miteinander kommunizieren konnten. Weil es AT&T angesichts seiner Monopolstellung im Telefonmarkt bis in die 1980er Jahre untersagt blieb, im IT-Bereich zu operieren, wurde Unix zunächst nicht kommerziell vertrieben, sondern mit Quellcode für einen symbolischen Betrag an Universitäten abgegeben, wo in der Folgezeit zentrale Systemerweiterungen wie etwa die Unterstützung der heute ubiquitär verwendeten TCP/IP-Protokollfamilie erarbeitet wurden. Die universitäre Unix-Entwicklung wurde durch die staatliche Advanced Research Projects Agency unterstützt, die bereits seit 1958 intensiv in Softwareprojekte öffentlicher Einrichtungen investierte (darunter ab 1968 auch das ARPANET), deren Ergebnisse nach akademischen Prinzipien publiziert wurden (Mowery/Langlois 1996).

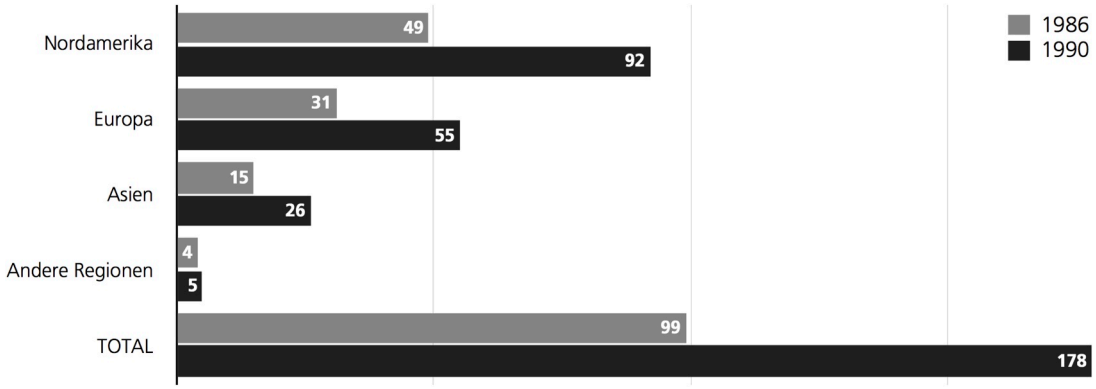
Zum einen wurde Software ab den 1960er Jahren zunehmend als eigenständiges Produkt wahrgenommen; zum anderen bildeten sich erste computerzentrierte Subkulturen in geschützten Nischen heraus, welche als Ausgangspunkt für spätere gefestigtere Gemeinschaften gelten können. Deren Beteiligte zeichneten sich durch eine hohe intrinsische Motivation aus, die Arbeitsgruppen waren informell strukturiert und die dort entwickelte Software stand meist zur freien Verfügung. Somit lassen sich diese Hobbyisten- und Hackergruppen in gewisser Hinsicht als frühe Varianten einer ‚peer production' in Benklers (2002) Sinne beschreiben. Neben Koordinationsproblemen bei steigender Projektgröße bestand aber nach wie vor die Gefahr der Proprietarisierung und Kommodifizierung ihrer Arbeitsergebnisse: Unix etwa wurde durch AT&T ab 1983 – sobald es kartellrechtlich möglich war – kommerzialisiert; der 1975 entstandene Homebrew Computer Club verlor seinen offenen Austauschcharakter, als mehr und mehr Teilnehmer eigene Firmen gründeten (darunter Apple Computer Inc.).

2.3 Institutionalisierung (1980/90er Jahre)

In den 1980er Jahren internationalisierte sich der Softwaremarkt zunehmend (Abb. 2) und die Hobbyistenkultur weitete sich nach Europa aus, was sich hierzulande unter anderem in der Gründung des Chaos Computer Clubs, dem Start der Fernsehsendung WDR Computerclub und der Einrichtung von „Computer-Centren" in Kaufhäusern niederschlug (Spiegel 50/1983: 172). Als Treiber für die freie Softwareent-

wicklung erwiesen sich die ab 1980 in Fachkreisen etablierenden neuartigen Formen elektronischer Vernetzung (darunter zuvorderst das Usenet), die den kostengünstigen Austausch von Daten über existente Telefonleitungen ermöglichten und so eine erste technische Lösung für die durch Frederick Brooks (1975) benannten Koordinationsprobleme in großen Projekten mit örtlich verteilten Entwicklern (Kap. 2.1) boten.

Abbildung 2: Umsätze mit Software und Services weltweit (Mrd. US-Dollar)



Eigene Berechnungen, Basis-USD-Werte für 1990. Datenquelle: Sayadian 1990.

Das Usenet war auch die Plattform, über die der MIT-Mitarbeiter Richard M. Stallman 1983 sein GNU-Projekt (rekursives Akronym für ‚GNU's not Unix') vorstellte, das als freies unixoides Betriebssystem eine Alternative zu proprietären Distributionen bieten wollte, deren Schutz und Ausschließbarkeit durch Gesetzesänderungen in den USA kurz zuvor noch einmal deutlich erhöht worden war (Menell 2002): „I cannot in good conscience sign a nondisclosure agreement or a software license agreement. So that I can continue to use computers without violating my principles, I have decided to put together a sufficient body of free software so that I will be able to get along without any software that is not free." (Stallman 1983)

Obgleich der angedachte komplett freie GNU Kernel per se bis heute nicht für den praktischen Einsatz geeignet ist, wurden im Kontext des Projekts früh Setzungen vollzogen, welche die freie Softwareentwicklung nach wie vor prägen: 1985 gründete Stallman die Free Software Foundation, die seitdem das ‚movement' an freiwilligen Entwicklern juristisch und infrastrukturell unterstützt. Zu ihren ersten Großspendern gehörten 1988 Sony (10.000 US-Dollar, technisches Equipment) und 1989 Hewlett-Packard (100.000 US-Dollar), die als Hardwarehersteller an günstig lizenzierbarer Software interessiert waren (Schwarz/Takhteyev 2010; O'Mahony 2005). Zudem fungiert die Free Software Foundation als Interessenvertretung für freie Software und aktualisiert die ‚Free Software Definition', die in erster Version lautete: „The word ‚free' [...] does not refer to price; it refers to freedom. First, the freedom to copy a program and redistribute it to your neighbors [...]. Second, the freedom to change a program, so that you can control it instead of it controlling you [...]." (FSF 1986: 8)

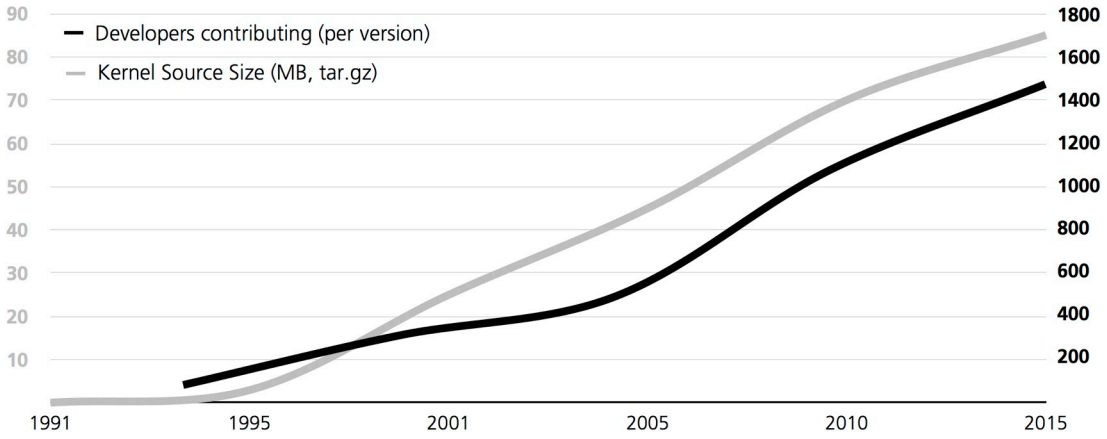
Die bedeutsamste Neuerung, die durch die Free Software Foundation initiiert wurde, bestand jedoch – statt GNU schlicht als *public domain* zu deklarieren – in der Ausarbeitung rechtlich belastbarerer Lizenzen, die erzwingen, dass auch Derivate freier Software stets frei bleiben müssen (‚Copyleft'). Nach der 1989 erstmals veröffentlichten GNU General Public License (GPL) dürfen entsprechend lizenzierte Programme modifiziert und weitergeben, aber nicht mit eigenen Einschränkungen belegt werden: „Each time you redistribute the Program [...], the recipient automatically receives a license from the original licensor [...]. You may not impose any further restrictions on the recipients' exercise of the rights granted herein." (FSF 1989) Ab 2001 waren Verstöße gegen die GPL Gegenstand mehrerer Gerichtsverfahren in Europa und Nordamerika, in denen Firmen wie Skype oder D-Link gezwungen wurden, fragliche Softwareelemente zu entfernen oder zur freien Verfügung zu stellen (Stiller 2011; Jaeger 2010). Osterloh/Rota (2007) und O'Mahony (2003: 1189) merken freilich an, dass für die Durchsetzung der in der GPL angelegten Reziprozitätsprinzipien nicht nur die offizielle Gerichtsbarkeit, sondern ebenso „the court of public opinion" im Usenet bzw. später im World Wide Web eine tragende Rolle gespielt hat.

Der Erfolg des GNU Projektes selbst blieb aufgrund seines Zuschnitts auf kostenintensive Workstations einerseits und seiner ideologischen Konnotationen andererseits begrenzt, da viele Entwickler auch für gut befundenen proprietären Code in ihre Architekturen integrieren wollten, was unter der GPL zunächst nicht möglich war (Weber 2000). Auf beide Problemstellungen bot das Linux Kernel Projekt eine Antwort: Linux wurde 1991 durch den Studenten Linus Torvalds als freier Betriebssystemkern für die günstigeren IBM-kompatiblen Mikrocomputer vorgestellt und war daher für eine größere Zahl an Entwicklern attraktiv. Zudem zeichnete sich Torvalds von Anfang an durch eine liberalere Haltung als die Free Software Foundation aus: „I think ideology sucks. This world would be a much better place if people had less ideology, and a whole lot more ‚I do this because it's fun and because others might find it useful, not because I got religion'." (Torvalds 2002) Zwar wurde auch der Linux Kernel 1992 unter die GPL gestellt, da aber der Kernel für ein lauffähiges Betriebssystem ohnehin mit weiteren Programmelementen verknüpft werden muss, schloss dies die Verwendung proprietärer Komponenten nicht grundsätzlich aus. Oft wird der Linux Kernel mit freien GNU Paketen kombiniert; in vielen Distributionen (z.B. Red Hat Enterprise Linux) kommen aber auch herstellerspezifische Module zum Einsatz.

Ein Grund für das Florieren des Linux-Projekts (Abb. 3) bestand nach den eher überschaubaren Erfolgen anbietergebundener Onlinedienste in den 1980er Jahren (z.B. Bildschirmtext, AppleLink, Compuserve) in der raschen Verbreitung des World Wide Web, das 1993 durch Tim Berners-Lee bzw. das CERN zur allgemeinen Nutzung freigegeben wurde. Sowohl der Zugriff auf GNU/Linux als auch die Beteiligung an entsprechenden Projekten und deren Koordination wurden durch die so möglich gewordenen neuen Austauschformen erheblich erleichtert. Überdies mussten sich freie

Softwareprojekte für Mikrocomputer zu dieser Zeit in der Praxis nur noch auf eine Systemumgebung konzentrieren: Während dieser globale Markt 1984 stark diversifiziert war (Commodore: 39 Prozent, IBM-PCs und Klone: 31 Prozent, Apple: 22 Prozent, Atari: 3 Prozent) lag der Anteil IBM-kompatibler Personal Computer 1995 bei über 90 Prozent (Reimer 2012). Nichtsdestotrotz blieb auch Linux zunächst ein lediglich in entsprechenden Expertenkreisen bekanntes Projekt.

Abbildung 3: Entwicklung der Linux Kernel Source Size



Datenquelle: Corbet 2009–2015, https://kernel.org (Stand: 6/2015).

Dies änderte sich mit dem vielrezipierten Buch „The Cathedral and the Bazaar“ (1999), das von Eric S. Raymond 1997 zuerst als Aufsatz vorgestellt wurde. Raymond (1998: 96) war von den Arbeitsweisen in Torvalds’ Umfeld fasziniert: „What I saw around me was a community which had evolved the most effective software-development method ever [...] without the theory or language to explain why the practice worked.“ Seine Kernthese: Während in klassischen Produktionsmodellen (*cathedral*) der Source Code eines Programms allenfalls für fertige Versionen publiziert wird und die Entwicklergruppen hierarchisch organisiert sind, sei der Quellcode in Projekten wie Linux oder dem durch ihn selbst initiierten Fetchmail stets einsehbar, ihre Gruppen seien horizontal strukturiert und geprägt durch modulare Selbstorganisation ohne zentrales Management (*bazaar*). Kritische Beobachter stellten allerdings früh fest, dass in beiden Fällen zwar viele Vorschläge aus der Community kamen, die finalen Änderungen aber nur durch eine Person – Torvalds oder Raymond – freigeben wurden (Connell 2000; Bezroukov 1999). Anders formuliert: „The only entity that can really succeed in developing Linux is the entity that is trusted to do the right thing. And as it stands right now, I’m the only person/entity that has that degree of trust.“ (Torvalds 1998: 36)

Mit GNU und Linux entstanden in den 1980/90er Jahren zwei Flaggschiffprojekte freier Softwareentwicklung, deren Erfolg durch die Effektivierung der Kommunikation im Usenet bzw. Web erheblich befördert wurde. In ihrem Kontext bildeten sich rechtliche Instrumente, welche die kollektiven Arbeitsresultate vor Proprietarisierung

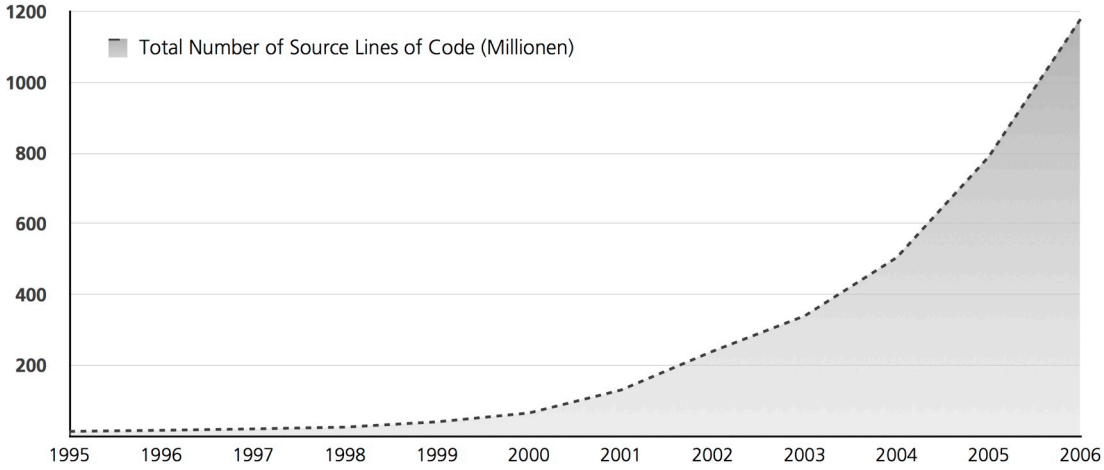
schützen, wie auch informelle Regeln heraus, deren Anerkennung sich im Onlinebereich unkomplizierter überprüfen ließ als zuvor. Daneben verfestigten sich erste anschlussfähige Narrative, welche die freie Softwareentwicklung als einen revolutionären, auf Peer-Review-Prozessen fußenden Produktionsmodus ohne Machtasymmetrien beschrieben und zeitweilig – trotz der aus dem akademischen Bereich bekannten Limitationen solcher Verfahrensweisen (Bezroukov 1999b) – ohne weitere Rückfragen sozialwissenschaftlich weiterverarbeitet wurden (z.B. Benkler 2002, 2006).

2.4 Wachstum – Diversifizierung (2000er Jahre)

Das Wachstum freier Softwareprojekte im nachfolgenden Jahrzehnt lässt sich neben der fortgesetzten Verbreitung des Internets vorrangig auf drei Dynamiken zurückführen. *Zum ersten* lagerte eine zunehmende Zahl an Firmen die Entwicklung von Softwareprodukten in den quelloffenen Bereich aus, darunter Netscape Communications als ein besonders früher und aufsehenerregender Fall: Nachdem es absehbar erschien, dass Microsoft den Netscape Navigator durch den in Windows integrierten Internet Explorer aus dem Markt drängen würde, kündigte das Unternehmen im Januar 1998 an, die Codebasis seines Browsers in das quelloffene Projekt Mozilla zu überführen, das bis 2003 durch Netscape/AOL über die Mozilla Organization personell und finanziell unterstützt wurde, die danach in die Mozilla Foundation überging. Dabei stand nicht zuletzt die Erschließung neuer Kundenkreise im Fokus: „By making our source code available to the Internet community, Netscape can expand its client software leadership by [...] building a community that addresses markets and needs we can't address on our own [...]." (Netscape 1998) Vier Jahre später wurde die Mozilla Application Suite veröffentlicht, aus der 2004 der Browser Firefox ausgekoppelt wurde, der zu einem weiteren Vorzeigeprodukt freier Softwareentwicklung werden sollte.

Angestoßen durch die Netscape-Entscheidung kam *zum zweiten* eine Gruppe um Eric Raymond Anfang 1998 zu dem Schluss, dass sich der politisch belegte Begriff ‚Free Software' für die weitere Verbreitung quelloffener Software in kommerziellen Kontexten als hinderlich erweisen könnte, schuf das neue Label ‚Open Source', das die Überlegenheit des Entwicklungsmodells betonen sowie gesellschaftsethische Aspekte ausblenden sollte (Raymond 1998b), und gründete mit Hilfe von Szeneprotagonisten wie Tim O'Reilly die Open Source Initiative (OSI). Allerdings unterstützte die Free Software Foundation (FSF) diese Kursänderung nicht. Zwar unterscheiden sich die ‚Open Source Definition' der OSI (Perens 1999) und die ‚Free Software Definition' der FSF kaum; trotzdem grenzt Richard Stallman (2002: 57) *free software* deutlich von *open source* ab: „For the Open Source movement, non-free software is a suboptimal solution. For the Free Software movement, non-free software is a social problem and free software is the solution." Aus diesem Dissens entwickelte sich ein Ausrichtungsstreit, der heute oft durch Hybridakronyme wie FOSS (‚Free & Open Source Software') oder FLOSS (‚Free/Libre Open Source Software') umgangen wird.

Abbildung 4: Wachstum populärer Open Source Projekte (inkl. Copy & Paste)



Datenquelle: Deshpande/Riehle 2008 (5122 meistverlinkte Projekte auf Ohloh).

Zu den Vermarktungsbemühungen der Open Source Initiative, welche die unternehmerischen Vorteile quelloffener Software betonten, kamen *zum dritten* die durch den allgemeinen Dotcom-Boom beförderten Börsenerfolge einiger *open source companies* im Jahr 1999 hinzu, darunter zuvorderst die linuxorientierten Hardwarehersteller VA Linux und Cobalt Networks sowie der Softwareanbieter Red Hat, der sich auf Linux-Architekturen für Unternehmen spezialisiert hat. Die Börsengänge dieser drei Firmen gehörten zu den erfolgreichsten Debüts aller Zeiten und erregten eine entsprechende mediale Aufmerksamkeit, die auf die Open Source Szene insgesamt abstrahlte (Gelsi 1999). Kurz danach beschrieb zum Beispiel der Spiegel (33/1999: 78, 30/2000: 55) Linux als „ernsthafte Konkurrenz zum Microsoft-Monopol" und brachte ausführliche Hintergrundberichte zu den Personalien Stallman, Raymond und Torvalds. Damit war ‚Open Source' als Schlagwort im öffentlichen Bewusstsein angekommen.

Diese ineinandergreifenden Dynamiken – die Überführung des Netscape Navigators in ein quelloffenes Projekt, die Etablierung des Labels ‚Open Source' und die anfänglichen Börsenerfolge linuxorientierter Firmen – führten im Verbund mit der weiteren Ausweitung des Marktes (globale Ausgaben für Software/Services 2005: 885 Mrd. US-Dollar, 2010: 1.092 Mrd. US-Dollar; UNCTAD 2012) zu einem fast exponentiellen Wachstum freier Softwareprojekte sowohl mit Blick auf ihre Größe (Abb. 4) als auch auf ihre Anzahl: Während Ende der 1990er Jahre einige hundert quelloffene Projekte existierten, waren 2008 auf der führenden Plattform SourceForge über 150.000 Projekte registriert. 2012 waren auf GitHub sowie SourceForge mehrere Millionen Repositorien und auf der Verzeichnisseite Ohloh (ab 2014: Open Hub) über 550.000 Projekte zu finden. Lediglich 47.000 dieser auf Ohloh katalogisierten Projekte wiesen jedoch Änderungen im vorangegangenen Jahr auf (Sands 2012). Der alleinige Verweis auf die Zahl registrierter Vorhaben sagt dementsprechend wenig aus.

Tabelle 2: Meistgenutzte quelloffene Softwarelizenzen 2015 (2010)

	Anteil 2015 (2010) in %	Ausrichtung	gebilligt durch	GPL kompatibel	Publikation
GNU Public License 2.0	23 (47)	strongly protective	OSI, FSF	•	1991
MIT License (X11)	22 (6)	permissive	OSI, FSF	•	1988
Apache License 2.0	16 (4)	permissive	OSI, FSF	•	2004
GNU Public License 3.0	10 (6)	strongly protective	OSI, FSF	•	2007
BSD License 2.0 (3-clause, n/r)	6 (6)	permissive	OSI, FSF	•	1999
Artistic License 1 / 2.0	5 (9)	[permissive]	OSI, [2.0: FSF]	– / [2.0: •]	2002
GNU Lesser GPL 2.1	5 (9)	weakly protective	OSI, FSF	•	1999
GNU Lesser GPL 3.0	2 (<1)	weakly protective	OSI, FSF	•	2007
Microsoft Public License	2 (2)	permissive	OSI, FSF	–	2007
Eclipse Public License	2 (<1)	permissive	OSI, FSF	–	2004
Code Project Open License	1 (3)	restrictive	–, –	–	2008
Mozilla Public License 1.1	<1 (1)	weakly protective	OSI, FSF	–	1999

Quelle: Black Duck Knowledgebase; Webseiten der OSI und FSF (Stand: 9/2015).

Vor dem Hintergrund der steigenden Zahl an Projekten, den Ausrichtungskonflikten in der Szene und der Definition eigener Lizenzmodelle durch Firmen und Stiftungen unterlag die quelloffene Softwareentwicklung in den 2000er Jahren einer Diversifizierung, die sich in einem inzwischen sehr breiten Bouquet an freien Lizenzen niederschlägt (Tab. 2): Neben strenge Copyleft-Lizenzen, die festlegen, dass auch Fortentwicklungen freier Software unter gleichen Bedingungen distribuiert werden müssen (*strongly protective*), sind Lizenzen getreten, welche die Einbindung freier Software in proprietäre Produkte erlauben, sofern ebendiese Komponenten quelloffen bleiben (*weakly protective*), oder die Publikation von Derivaten unter restriktiveren Bedingungen ermöglichen (*permissive*). Diese Vielfalt erweitert die strategischen Optionen für kommerzielle Stakeholder (Lerner/Schankerman 2010; Lerner/Tirole 2005): Nachdem 2007 die GPL in dritter Version publiziert und zuvor ausschöpfbare Lücken geschlossen worden waren, tauschte etwa Apple die GNU Compiler-Sammlung GCC in seiner Entwicklungsumgebung Xcode durch eine Lösung mit freizügiger Lizenz aus; Google entschied sich im Falle des mobilen Betriebssystems Android von vornherein dazu, den Großteil des Codes unter die permissive Apache License 2.0 zu stellen.

Daneben lässt sich in den letzten 15 Jahren in zweierlei Hinsicht eine „corporatization of open source“ (Hogan 2009) beobachten: Zum einen werden zentrale Projekte wie der Linux Kernel, Mozilla Firefox, Apache HTTP, die Programmierumgebung Eclipse und die Cloud-Architektur OpenStack durch Spenden von Unternehmen mitfinanziert oder operieren wie die HTML Rendering Engine WebKit (Apple) und die Android Plattform (Google) unter der Federführung kommerzieller Anbieter (Fitzgerald 2006). Zum anderen speist sich die Entwicklerbasis großer Open Source Projekte zu-

nehmend aus Unternehmenskontexten: Kolassa et al. (2014) kommen in ihren Analysen zum Linux-Kernel-Projekt und 5000 weiteren auf Open Hub registrierten Vorhaben zu dem Schluss, dass zwischen 2000 und 2011 mehr als 50 Prozent aller Beiträge in der westlichen Kernarbeitszeit (Montag bis Freitag, 9 bis 17 Uhr) geleistet wurden; die Linux Foundation beobachtet, dass der Anteil unbezahlter, nicht unternehmensaffiliierter Beiträger an der Kernel-Entwicklung kontinuierlich abnimmt (2009: 18 Prozent; 2014: 12 Prozent), während der Anteil an Aktualisierungen von Entwicklern aus den mittlerweile rund 250 beteiligten Firmen weiter anwächst (Corbet et al. 2015).

In die Linux-Kernel-Entwicklung involviert sind neben Linux Distributoren wie Red Hat auch marktbestimmende IT-Konzerne. IBM (Umsatz 2014: 93 Mrd. US-Dollar) zeigt sich als einer der führenden Anbieter von Server-Hardware an der Entwicklung des Linux Kernels überaus interessiert und annoncierte zuletzt 2013 ein Investment über ca. 1 Mrd. US-Dollar in Open Source Technologien (IBM 2013). Aus ähnlichen Gründen investiert Samsung (Umsatz 2014: 188 Mrd. US-Dollar) in das Projekt: Die Firmware zahlreicher Produkte des Unternehmens fußt auf Linux; zudem hat Samsung 2011 zusammen mit Intel und weiteren Partnern die Entwicklung des linuxbasierten Betriebssystems Tizen angestoßen, um sich im Bereich der Mobile Devices und Medienabrufgeräte langfristig von Android absetzen zu können. Vergleichbare Interessen lassen sich für andere Hersteller identifizieren. Insgesamt zeichnen sich Entwickler aus den vier Unternehmen Intel, Red Hat, Samsung und IBM seit Version 3 für rund 25 Prozent der Änderungen im Linux Kernel verantwortlich (Tab. 3).

Tabelle 3: Änderungen im Linux Kernel nach Unternehmen/Organisationen

	2013–2014 (R 3.11–3.18)	**2011–2013** (R 3.0–3.10)	**2010–2012** (R 2.6.36–3.2)	**2005–2009** (R 2.6.11–2.6.30)
unabhängig	**12,4 %**	13,6 %	16,2 %	18,2 %
nicht identifizierbar	**4,9 %**	3,3 %	4,3 %	7,6 %
Intel	**10,5 %**	8,8 %	7,2 %	5,3 %
Red Hat	**8,4 %**	10,2 %	10,7 %	12,3 %
Linaro	**5,6 %**	4,1 %	0,7 %	n.a.
Samsung	**4,4 %**	2,6 %	1,7 %	n.a.
IBM	**3,2 %**	3,1 %	3,7 %	7,6 %
SUSE/Novell	**3,0 %**	3,5 %	4,3 %	7,6 %
Consultants	**2,5 %**	1,7 %	2,6 %	2,5 %
Texas Instruments	**2,4 %**	4,1 %	3,0 %	n.a.
Vision Engraving Systems	**2,2 %**	2,3 %	n.a.	n.a.
Google	**2,1 %**	2,4 %	1,5 %	0,9 %
andere Unternehmen	**38,4 %**	40,3 %	44,8 %	38,0 %
Intel/Red Hat/Samsung/IBM TOTAL	**26,5 %**	24,7 %	23,3 %	25,2 %

Eigene Berechnungen. Datenquelle: Corbet et al. 2009–2015.

In den letzten zwei Jahrzehnten konnte sich die quelloffene Entwicklung folglich als *Methode* zunehmend in der Softwarebranche etablieren. Sie hat dabei aber ihre Formatierung als *Gegenentwurf* zur proprietären Softwareentwicklung weitgehend verloren. Zwar lassen sich nach wie vor Liebhaberprojekte wie die vergleichsweise unregelmäßig aktualisierten Linux-Distributionen Arch, Dragora oder Parabola finden, die sich an den generischen Maximen freier Software ausrichten. In die meisten aktiven Open Source Projekte sind inzwischen jedoch etablierte Unternehmen involviert, die diese Kontexte nutzen, um außerhalb formaler Kooperationsbeziehungen mit externen Programmierern zu kollaborieren und so ihre ansonsten nach außen eher abgeschotteten Forschungs- und Entwicklungsaktivitäten durch „kontrollierte Öffnungen an den Rändern" (Dolata 2015: 17) zu erweitern. Im Unterschied zu den historischen Beispielen für ‚collective invention' (Tab. 1) werden die Resultate dieser Kollaboration heute jedoch durch rechtlich belastbare Lizenzen geschützt, die sich zwar in unterschiedlichem Maße an den originären Reziprozitätsprinzipien freier Software ausrichten, aber stets die Möglichkeit zur Weitergabe und Modifikation einräumen.

Die konzeptionellen Ursprünge dieser Lizenzformen lassen sich in den frühen 1980er Jahren verorten, als die Zeit, in der Software eher als *research tool* und weniger als *commodity* wahrgenommen wurde, noch nicht weit zurücklag und neue elektronische Vernetzungstechnologien zugleich erstmals die ortsunabhängige Koordination größerer Projektkontexte ohne größeren infrastrukturellen Aufwand ermöglichten. In dieser Anfangsphase onlinebasierter freier Softwareentwicklung in einer von Marktmechanismen weithin abgelösten Nische lassen sich wohl am ehesten verbreitete Beispiele für eine von korporativen Interessen abgekoppelte ‚commons-based peer production' finden. Ab Mitte der 1990er Jahre wurden günstig lizenzierbare quelloffene Softwarekomponenten freilich zu Eckpfeilern einer internetzentrierten Startup-Szene und in den Folgejahren stellen sich auch klassische Unternehmen auf die institutionellen Rahmenbedingungen freier Softwareentwicklung ein. Insofern beschreibt der Blogger Mike Bulajewski (2011) das Bild von Open Source Projekten als Gemeinschaften „of volunteer programmers collaborating together in a gift economy with no financial incentives" zurecht als Illusion. Vielmehr fußt die Vitalität vieler Projekte wesentlich auf dem Engagement großer Konzerne, die ihre Ressourcen erwartungssicherer einbringen können, als dies freiwilligen Einzelentwicklern meist möglich ist.

3 Open Source und kommerzieller Markt

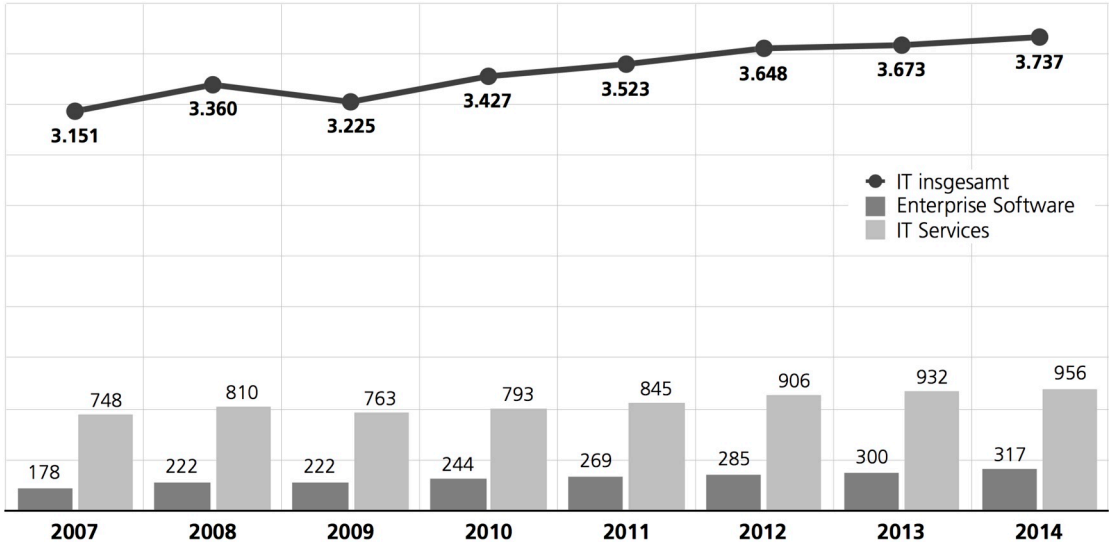
Quelloffene Softwareprojekte und kommerzieller IT-Markt sind heute eng ineinander verwoben (Westenholz 2012; Lerner/Schankerman 2010). Open Source Architekturen nehmen im Bereich der basalen informationstechnischen Infrastrukturen signifikante Marktanteile ein; auch und gerade in Unternehmen werden freie und proprietäre Programmelemente häufig in Kombination miteinander eingesetzt; kommerziell vertrie-

bene Softwarepakete tragen oft Open Source Komponenten in sich et vice versa; Support- und Integrationsdienstleistungen um quelloffene Software sind – wenn auch weniger für dezidierte *open source companies* als für etablierte Anbieter – zu einem einträglichen Geschäftsfeld geworden. Überdies zeigt sich, dass die meisten marktrelevanten Open Source Projekte mittlerweile nicht mehr ohne das finanzielle wie personelle Engagement mittlerer und großer Unternehmen auskommen.

3.1 Marktrelevanz quelloffener Software

Der allgemeine Markt für Software und Services hat sich in den letzten Jahren erneut ausgeweitet (Abb. 5). Die globalen Erlöse mit *packaged software* beliefen sich 2014 auf 400 bis 450 Mrd. US-Dollar (Gartner 2015; IDC 2015) und die größten softwarezentrierten Firmen machen inzwischen Jahresumsätze, die mit klassischen produzierenden Unternehmen wie Bosch (2014: 49 Mrd. Euro) oder Airbus (2014: 61 Mrd. Euro) vergleichbar sind (Tab. 4). Der Großteil der Umsatzes wird dabei im Enterprise Bereich generiert: Microsoft (2015) setzte 2014 rund 19 Mrd. US-Dollar mit Consumer Software und 42 Mrd. US-Dollar mit Unternehmenssoftware um; IBM (2015) generierte 2014 über 90 Prozent seines Softwareumsatzes mit Middleware und Betriebssystemen; SAP, Oracle, EMC, Symantec, CA und Ericsson operieren ebenfalls primär im Unternehmensbereich. Für dieses Segment diagnostizieren Marktforscher mittlerweile einen „widespread use of open-source technology in mission-critical IT portfolios“ und sehen den Grund dafür nicht nur in Kostenvorteilen, sondern auch in der Anpassbarkeit quelloffener Architekturen (Driver 2014; Aponovich et al. 2014).

Abbildung 5: Weltweite Ausgaben für IT in Mrd. US-Dollar 2007–2014



Quelle: Gartner Inc. Press Releases (Stand: 5/2015); IT gesamt: Hardware und Devices, Rechenzentren, Software, IT-Services, Telekommunikation.

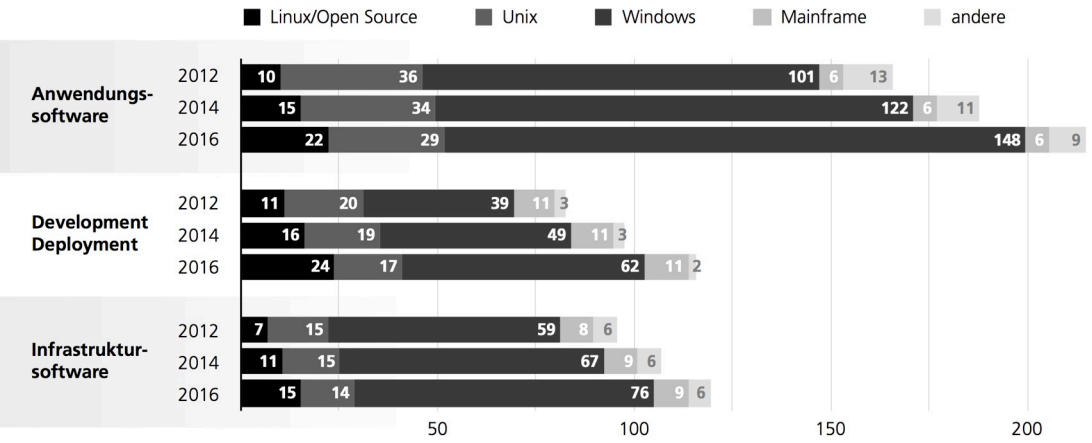
Tabelle 4: Weltweit führende Softwarefirmen 2012 und 2014 (in Mrd. US-Dollar)

Unternehmen (Rang 2012)	Umsatz Software 2012*	Gesamt-umsatz 2012*	Umsatzanteil Software 2012	Umsatz Software 2014**	Gesamt-umsatz 2014**	Ausgaben für FuE 2014***
(1) Microsoft	58,4	72,9	**80 %**	60,8	86,8	11,4
(2) IBM	28,8	104,5	**28 %**	25,4	92,8	5,5
(3) Oracle	27,7	37,3	**74 %**	29,2	38,2	5,1
(4) SAP	16,6	21,3	**78 %**	15,9	18,9	2,5
(5) EMC (mit VMware)	9,3	21,7	**43 %**	n.a.	24,4	2,9
(6) Ericsson	8	34,9	**23 %**	6,3	26,2	4,2
(7) Symantec	6,4	6,8	**94 %**	n.a.	6,7	1
(8) Hewlett-Packard	5,5	119,2	**5 %**	3,9	111,5	3,4
(9) Adobe Systems	4,3	4,4	**98 %**	n.a.	4,1	0,8
(10) CA Technologies	4,3	4,6	**92 %**	4,1	4,5	0,6
(28) Apple	1,6	164,7	**1 %**	n.a.	182,8	6
(52) Google	0,8	50,1	**2 %**	n.a.	66	9,8

** PWC 2014; ** Form 10-K/Jahresberichte; *** FuE: Forschung und Entwicklung.*

Ein Marktausblick der International Data Corporation (IDC), der sich mit der Frage beschäftigt, wie sich der globale Gesamtumsatz mit Anwendungs-, Administrations- und Systemsoftware nach Betriebsumgebungen aufteilt (Abb. 6), führt zwar zunächst die herausgehobene Stellung vor Augen, die Microsoft Windows als *operating environment* für Standardapplikationen 2014 mit 64 Prozent noch immer einnimmt. Und ebenso bleibt Windows im Bereich der Administrations- und Systemsoftware mit einem Anteil von über 50 Prozent laut dieser Studie die präferierte Einsatzumgebung, wobei 2016 immerhin 16 Prozent der Umsätze mit linuxorientierten Programmen generiert werden sollen. Allerdings beziehen sich diese Projektionen ausschließlich auf marktlich gehandelte Standardsoftware und reflektieren weder das umsatzträchtige Feld der IT-Services noch den Einsatz kostenfreier Werkzeuge und Frameworks.

Abbildung 6: Prognostizierter Umsatz nach Betriebsumgebungen (Mrd. US-Dollar)



Datenquelle: International Data Corporation 2012.

Auf Trackingplattformen wie StatCounter oder W3techs, die öffentlich einsehbare Kenndaten von onlinegeschalteten Computern aggregieren, lässt sich indes nachvollziehen, dass Open Source Software inzwischen in vielen Marktsegmenten von signifikanter Bedeutung ist (Tab. 5). Dies gilt (abgesehen von Android) weniger für den Consumer Bereich und *client computer* (‚sichtbare Arbeitsrechner') als vielmehr für das Feld der serverseitigen Infrastrukturen: Während auf einem Personal Computer nach wie vor nur in Ausnahmefällen GNU/Linux läuft, Microsoft Office noch immer De-Facto-Standard für Textverarbeitungs-, Tabellenkalkulations- und Präsentationsprogramme ist und sich der globale Marktanteil von Mozillas Firefox Browser mit der Verbreitung von Google Chrome reduziert hat, nimmt Linux als Betriebssystem für öffentliche Server stabil zwei Drittel des Marktes ein; im Bereich des relationalen Datenbankmanagements hat sich der Anteil von Lösungen mit quelloffener Lizenz zuletzt auf knapp 45 Prozent erhöht; das Feld der HTTP Server wird bereits seit den 1990er Jahren von Open Source Architekturen beherrscht; und unter den verbreiteten Content Management Systemen hat WordPress seinen Vorsprung weiter ausgebaut.

Tabelle 5: Geschätzte weltweite Marktanteile quelloffener Software (in Prozent)

	Open Source	2010	2015	Mitbewerber	2010	2015
Betriebssystem Personal Computer (a)	GNU/Linux	1	1,6	MS Windows Apple Mac OS X	94 5	91 7
Betriebssystem Mobile Devices (b)	Android	11	66	Apple iOS Symbian/Nokia OS Blackberry Windows Phone	30 33 14 —	19 3 1 2
Webbrowser Desktop (c)	Mozilla Firefox	31	18	Google Chrome MS IE Apple Safari	14 47 5	57 17 4
Office Suites in Organisationen [2013] (d)	Open Office / Libre Office	2		MS Office (+ SaaS) Google Docs	91 5	
Betriebssystem öffentliche Server (e)	Linux (mit unix-oiden Systemen)	69	67	MS Windows	31	33
Webserver [aktive Sites] (f)	Apache Nginx	72 4	56 26	Microsoft IIS Google Servers LiteSpeed	21 1 1	13 1 2
Datenbankmanagement (g)	OSS Lizenz (z.B. MongoDB)	35 (2012)	44	Kommerzielle Lizenz (z.B. Oracle)	64 (2012)	56
Web Content Management System (h)	WordPress Joomla Drupal Typo 3	51 12 7 4	59 7 5 2	Blogger (Google) Bitrix vBulletin Shopify	2 — 8 —	3 1 1 1

Datenquelle (Stand: 9/2015): (a) NetApplications; (b, c) StatCounter; (d) Forrester 2013; (e, f, h) W3techs; (g) DB-Engines.

Erhebungsdaten des Consultingunternehmens Forrester (2014) zeigen darüber hinaus auf, dass 2014 rund 80 Prozent der Softwareentwickler in Unternehmen weltweit auf Open Source Code zurückgegriffen haben – und zwar vorrangig im Datenbank-, Server- und Systemmanagement sowie in der Erstellung mobiler Anwendungen. Die gleiche Studie spricht quelloffenen Lösungen auf den Feldern des Cloud Computings (73 Prozent) und der Sicherheitsarchitekturen (59 Prozent) Marktführerschaft zu. Zudem kommt eine Erhebung unter Open Source Nutzern zu dem Ergebnis, dass universitäre sowie schulische Organisationen auf einigen Einsatzfeldern beinahe vollständig auf quelloffene Architekturen zurückgreifen (BlackDuck/NorthBridge 2015; ähnlich: Ghosh et al. 2006). Dem Analysten Jeffrey Hammond (2014: 21. Min.) zufolge geht mit dem korporativen Einsatz von Open Source Software freilich nicht nur die Hoffnung auf eine Kostenreduktion, sondern auch auf einen vereinfachten Anschaffungsprozess einher: „[Developers] are basically saying: Hey, we're trying all the open source alternatives first, because it's so painful dealing with our procurement department, that we'll only go commercial if we can prove that the open source stuff can't work." Diesen Spielraum haben Entwickler, da viele Organisationen bislang keine Richtlinien für den Einsatz buchhalterisch zunächst kostenfreier quelloffener Software ausgearbeitet haben. Auch wenn sich der Programmcode unentgeltlich nutzen lässt, schmälert dies freilich nicht die internen Trainings- und Integrationskosten, auch weil Open Source Produkte oft keiner Support- und Gewährleistungspflicht unterliegen.

Zusammengenommen sprechen die reflektierten Daten und Statistiken für die u.a. durch die Marktforschungsunternehmen Gartner und Forrester vertretende Einschätzung, dass quelloffenen Softwarearchitekturen inzwischen vor allen Dingen im Bereich der grundlegenden kommunikations- und informationstechnischen Infrastrukturen eine hohe Marktrelevanz zugesprochen werden kann und es daher insbesondere für größere privatwirtschaftliche sowie staatliche Organisationen zunehmend unmöglich wird, nicht auf Open Source Architekturen zurückzugreifen: „By 2016, at least 95 percent of IT organizations will leverage nontrivial elements of open-source software technology [...] including cases where they might not be aware of it." (Driver 2013: 2. Min.) Vor diesem Hintergrund kommen auch schon lange tätige Hersteller proprietärer Softwareprodukte wie Microsoft, IBM, Adobe, Oracle oder SAP nicht umhin, sich auf dieses veränderte Umfeld einstellen.

3.2 Involvement etablierter Anbieter

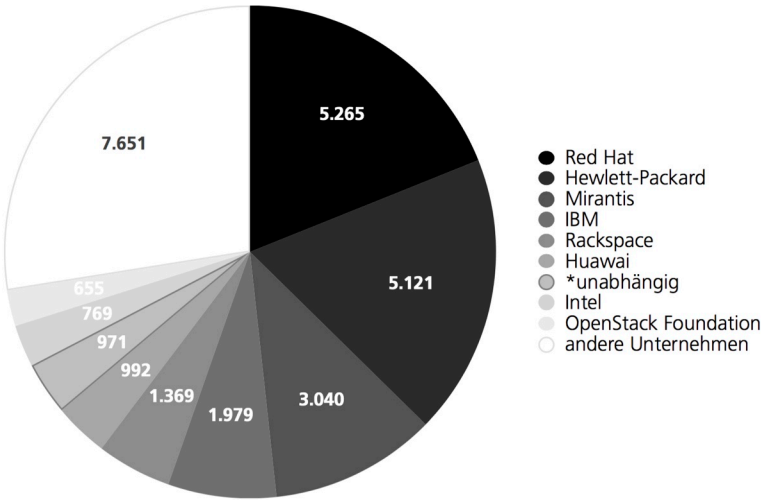
Dementsprechend sind die meisten großen Softwareanbieter heute intensiv in quelloffene Projekte involviert. *Microsoft* – das Unternehmen, das Open Source lange als „intellectual-property destroyer" bezeichnet hat (Leonard 2001) und nach wie vor die „competitive advantages" hervorhebt, die aus geschützter interner Entwicklungsarbeit resultieren (Microsoft 2015: 12) – betreibt mit CodePlex seit 2006 eine Webplattform für Open Source Vorhaben, hat 2012 die Tochterfirma MS Open Technologies lan-

ciert, gehörte zwischenzeitig zu den Top-Beiträgern in der Linux-Kernel-Entwicklung (Corbet 2012) und hat in den letzten Jahren u.a. das Softwareframework .NET unter quelloffene Lizenz gestellt. Dabei geht es Microsoft vor allem anderen darum, angesichts zunehmend heterogener IT-Infrastrukturen im Enterprise Bereich übergreifende Standards abzusichern und die Kompatibilität eigener Produkte zu erhöhen: „At times, we make select intellectual property broadly available at no or low cost to achieve a strategic objective, such as promoting industry standards, advancing interoperability, or attracting and enabling our external development community." (Microsoft 2015: 13) Microsofts Linux Engagement etwa lässt sich primär auf die Integration von Treibern für die Virtualisierungstechnik Hyper-V als Teil von Windows Server zurückführen (McAllister 2013); zusammen mit Google, IBM und anderen Unternehmen hat Microsoft 2014 unter dem Eindruck des schwerwiegenden OpenSSL-Fehlers „Heartbleed" zudem die Core Infrastructure Initiative (2015) geformt, um „open source projects that are in the critical path for core computing functions" langfristig zu fördern.

Welche Anteile ihrer Ausgaben für Forschung und Entwicklung Microsoft und andere marktführende Unternehmen in Open Source Projekte investieren, lässt sich aus den jährlichen Geschäftsberichten nicht herauslesen und kaum gesondert abschätzen, da quelloffene Komponenten heute für viele herstellerspezifische Softwareprodukte eine tragende Rolle spielen. *Apples* Betriebssystempakete liefern dafür ein anschauliches Beispiel: Sowohl Mac OS X als auch iOS basieren auf dem freien Betriebssystemkern Darwin und tragen mehr als 200 weitere Open Source Komponenten mit sich, so zum Beispiel die HTML Rendering Engine WebKit, die in vielen verbreiteten Produkten Anwendung findet (z.B. von Sony, Google, Amazon). Korrespondierend dazu speist sich die Entwicklerbasis von Darwin fast ausschließlich aus dem Unternehmen Apple; der überwiegende Teil der Änderungen im WebKit Projekt stammte zwischen 2002 und 2013 von bei Google oder Apple angestellten Mitarbeitern (Bitergia 2013).

Auch das Unternehmen *IBM*, das seit jeher den Großteil seines Umsatzes mit Hardware- und Softwarelösungen für Geschäftskunden verdient, beteiligt sich intensiv an Open Source Projekten: Bereits zur Jahrtausendwende investierte IBM mehrere 100 Mio. US-Dollar in Linux-Entwicklungsprogramme und portierte seine Software für GNU/Linux (Capek et al. 2005). Zum ersten verfolgte IBM damit das Ziel, Microsofts Dominanz im Bereich der Enterprise Software entgegenzusteuern: „The arrival of Linux and the open source movement was well-timed for IBM [...]. Here is a software development and distribution model that potentially threatens the core of the Microsoft business model." (Software Magazine 6/2001: 4) Zum zweiten ging es IBM wie schon im Falle von FORTRAN (Kap. 2.1) darum, das Servicegeschäft um proprietäre Technologien im Verbund mit quelloffener Software auszubauen und die Verbreitung IBM-naher Standards zu erhöhen. Und zum dritten konnte IBM als Arbeitgeber in Entwicklerkreisen früh von dem Image „[of] being a patron for industry-wide open source efforts" profitieren (Coté et al. 2007: 4; Fitzgerald 2006).

Abbildung 7: Commits in OpenStack – Release Liberty (2015)



Datenquelle: http://stackalytics.com.

Inzwischen ist IBM in über 100 Open Source Projekte involviert, darunter auch die seit 2010 entwickelte Cloud Computing Architektur OpenStack, welche seit 2013 die Grundlage für sämtliche IBM Cloud Services bietet. „IBM clearly wants to influence OpenStack's technological direction and efforts to develop industry standards for cloud computing, which is still a relatively immature architecture." (Gonsalves 2013) Neben IBM sind auch die multinationalen IT-Konzerne *Intel* und *Hewlett-Packard* intensiv an der Entwicklung von Open Stack beteiligt (Abb. 7). Ihr Engagement resultiert jedoch ebenfalls nicht aus Idealismus, sondern aus unternehmerischem Kalkül: „Such actions are comparable to giving away the razor (the code) to sell more razor blades (the related consulting services [...])." (Lerner 2012: 43) Im Falle von OpenStack erleichtert die permissive Lizenzierung des Projektes zudem die Überführung in herstellerspezifische Lösungen wie die Enterprise Cloud Plattform HP Helion.

Aus vergleichbaren Gründen beteiligen sich andere Firmen an Open Source Projekten: *Oracle* verdient sein Geld seit den 1970er Jahren mit Datenbanklösungen für staatliche und privatwirtschaftliche Einrichtungen, ist in diesem Kontext auf die Interoperabilität seiner Komponenten mit quelloffenen Infrastrukturen angewiesen und verfolgt mit dual lizenzierten Produkten wie MySQL überdies das Geschäftsmodell, erweiterte Enterprise Versionen von Open Source Software inklusive Supportdienstleistungen zu verkaufen. Auch *SAP* ist als führender Anbieter von Enterprise Resource Planning Systemen darauf bedacht, die Entwicklung quelloffener Software zu beobachten und mitzugestalten, um die Kompatibilität eigener Produkte abzusichern sowie „Kunden aufzuzeigen, dass sie ihr Open-Source-Know-how auch in SAP-Umgebungen sinnvoll einsetzen können" (Spies 2010). *Adobe* hingegen sieht sein Open Source Involvement vorrangig „as part of a strategy to assist and grow the developer community in line with the company's products and markets" (Germain 2014).

Eine spezielle Variante korporativen Open Source Engagements stellt die Entwicklung des linuxbasierten Betriebssystems Android durch die Open Handset Alliance seit 2007 dar: Beworben als „open-source software stack [...] to make sure there was no central point of failure, where one industry player could restrict or control the innovations of any other“ (http://source.android.com), und in der Literatur oft in eine Linie mit Linux oder Apache gestellt (Herstatt/Ehls 2015: XVII), wird das Projekt de facto alleinig von *Google* gesteuert. „Google largely follows the ‚cathedral‘ model of software production [...]. Because it fully controls the development of the OS, Google can determine the technological specifications to which Android partners must abide.“ (Spreeuwenberg/Poell 2012) Mit der Initiierung des Android Projekts ging es Google (2015) mit offenkundigem Erfolg vor allen Dingen darum, den nahtlosen Zugriff auf eigene Dienste und Plattformen wie Google Play auf möglichst vielen Geräten zu ermöglichen: Während Google 2007 knapp 99 Prozent seines Umsatzes (16,6 Mrd. US-Dollar) mit Werbung generierte, war der Verkauf digitaler Inhalte 2014 für 11 Prozent des Jahresumsatzes (66 Mrd. US-Dollar) verantwortlich.

Davon abgesehen lässt sich das Involvement großer Softwarekonzerne in die quelloffene Entwicklung zum einen aus der Notwendigkeit ableiten, die Position eigener Lösungen in einem zunehmend von Open Source Komponenten durchdrungenen Umfeld abzusichern. Zum anderen ist es traditionell primär im Enterprise Bereich tätigen Anbietern wie HP, IBM und Oracle in den letzten Jahren gelungen, ein Dienstleistungsgeschäft um Open Source Architekturen und deren Integration mit proprietären Komponenten zu etablieren. Vor allen Dingen für kleinere Softwarefirmen dient das Engagement in Open Source Projekten oder die Gründung eigener Vorhaben zudem als „marketing tool to increase brand recognition“ (Dahlander/Magnusson 2008: 638): Gelingt es, das eigene Unternehmen mit prominenten Projekten zu assoziieren und davon ausgehend Fachbücher oder Konferenzbeiträge zu lancieren, trägt dies gegenüber Kunden und externen Entwicklern zur Steigerung des Firmenrenommees bei.

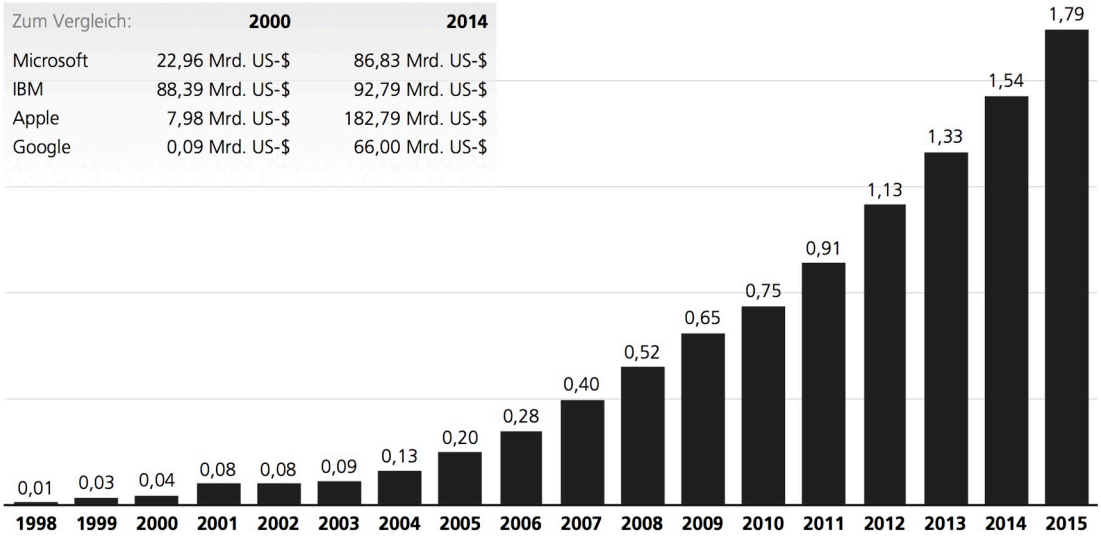
3.3 Open Source Companies

Neben den genannten hybriden Verwertungsmodellen etablierter Anbieter haben sich Ende der 1990er Jahre eine Reihe reiner *open source companies* herausgebildet, die ihr Kernprodukt – den Softwarecode – kostenfrei abgeben und versuchen, mit auf den Unternehmenseinsatz zugeschnittenen Editionen im Verbund mit Dienstleistungen ein Geschäft aufzubauen. Mit Ausnahme von Red Hat sind die meisten dieser im Fahrwasser des New Economy Hypes gegründeten Open Source Firmen allerdings inzwischen wieder eingegangen oder von größeren Unternehmen übernommen worden: Sleepycat (Datenbankmanagement) etwa gehört (wie MySQL AB) seit 2006 zu Oracle; JBoss (Middleware) wurde durch Red Hat aufgekauft; SpringSource (Java Framework) ist seit 2008 Teil von VMware. Pointiert zusammengefasst: „[...] beyond Red Hat the effort [of commercializing open source] has largely been a failure

from a business standpoint. Consider that the ‚support' model has been around for 20 years, and other than Red Hat there are no other public standalone companies that have been able to offer an alternative to their proprietary counterpart." (Levine 2014)

Red Hat bietet inzwischen ein breites Portfolio an unternehmenszentrierter Software inklusive Support auf Subskriptionsbasis an – von Cloud Computing Architekturen bis hin zu Lösungen für die Administration und Entwicklung. Den Großteil seines jährlichen Umsatzes (2014 geschätzte 87 Prozent) generiert das Unternehmen jedoch mit seinem Betriebssystem Red Hat Enterprise Linux, das durch integrierte Systemmanagement-Dienste die Verwaltung von IT-Netzwerken in Großorganisationen erleichtert. Zu seinen prominentesten Kunden zählen das US Department of Defense, das Filmstudio DreamWorks und die New Yorker Börse; ferner unterhält Red Hat Partnerschaften mit Amazon, HP, Cisco, IBM und weiteren führenden Unternehmen. Nach Schätzungen der IDC hielt Red Hat 2014 einen Anteil von 64 Prozent auf dem weltweiten Markt für kommerzielle Linux-Distributionen (Ante 2014).

Abbildung 8: Red Hat Jahresumsätze (Mrd. US-Dollar; fiscal years ending 28. Feb.)



Datenquelle: Jahresberichte der Unternehmen.

Der in dieser Form singuläre Erfolg von Red Hat (Abb. 8) kann auf die generell zunehmende Adaption von Linux im Enterprise Bereich zurückgeführt werden, die sich nicht zuletzt mit einer erhofften Vermeidung von *vendor lock-ins* und Einsparungspotentialen begründen lässt, aber erwartungssichere Supportleistungen keineswegs obsolet macht. In diesem Kontext konnte Red Hat gegenüber späteren Mitbewerbern einen First Mover Vorteil ausspielen, da Red Hat Linux und die Verwaltungsplattform Red Hat Network bereits ab 2000 ein stetig weiterentwickeltes Gesamtsystem bildeten und das Unternehmen frühzeitig Kooperationen mit marktdominanten Hard- und Softwareanbietern eingegangen ist (West/Dedrick 2001). Neben Red Hat und

kleineren Anbietern wie SUSE (Umsatz 2013: 200 Mio. US-Dollar) und Canonical (Umsatz 2013: 66 Mio. US-Dollar) wird der Weltmarkt für Linux (Hardware, Software, Services), dessen Gesamtumsatz für 2016 auf 70 Mrd. US-Dollar (2012: 35 Mrd. US-Dollar) geschätzt wird (Gillen 2013), durch die entsprechenden Divisionen größerer Konzerne bestritten, welche intensiver in Forschung und Entwicklung bzw. Vermarktung investieren können, als dies etwa Canonical oder Red Hat möglich ist.

Außerhalb der Linux-Entwicklung sind im Open-Source-Umfeld in den letzten zehn Jahren eine Vielzahl neuer Startup-Firmen entstanden, die zwar mitunter beträchtliche Investorengelder einwerben konnten, aber oft nicht profitabel operieren. Das Unternehmen Hortonworks (2015) beispielsweise, das auf der Basis von Apache Hadoop Lösungen zur distribuierten Verarbeitung großer Datenmengen anbietet und 2011 bis 2014 knapp 250 Mio. US-Dollar an *investor fundings* erhalten hat, wies für 2014 einen Umsatz von 46 Mio. US-Dollar bei Betriebskosten von 135 Mio. US-Dollar aus; die seit 2011 an der Börse notierte Firma Jive Software (2014), die auf Wissensmanagement-Architekturen für den geschäftlichen Bereich spezialisiert ist, machte 2014 einen Verlust von 55 Mio. US-Dollar (2013: 75 Mio. US-Dollar). Daneben existieren eine Reihe privat gehaltener open-source-affiner Startups, die ihre Kennzahlen nicht öffentlich ausweisen, aber ebenfalls zumeist (noch) nicht selbsttragend sind, darunter etwa SugarCRM (Kundenmanagement) oder Alfresco (Dokumentenverwaltung).

In ihrer Außendarstellung verzichten viele dieser Firmen mittlerweile allerdings zugunsten klassischer Nutzenversprechen (z.B. Reliabilität, Kostenvorteile) auf ‚Open Source' als primäres Differenzierungsmerkmal. Wie Magnus Bergquist und Kollegen (2012) in qualitativen Interviews mit Entwicklern in *open source companies* festgestellt haben, zeichnen sich solche Unternehmen oftmals durch eine nur noch geringe Loyalität gegenüber den Reziprozitätsidealen freier Software aus: „At the end of the day the company must earn money to survive. Richard Stallman has a very idealistic view of the world, which is admirable. But if one considers it from a business perspective one realizes that it is not feasible in practice." (Service Provider, in ebd.: 8) Stattdessen sind es heute vor allem große Anbieter wie IBM („Open Source & Standards are key to making our planet smarter") oder Microsoft („Openness builds bridges between platforms and people"), die in ihrer Öffentlichkeitsarbeit auf ausgewählte Maximen freier Softwareentwicklung verweisen (IBM 2015b; Microsoft 2015b).

3.4 Patronage und Spendenfinanzierung

Imagepflege ist allerdings nur einer der Gründe, warum führende IT-Konzerne neben ihrem Involvement in die Code-Entwicklung als Sponsoren und Partner für ein breites Portfolio an Open Source Vorhaben auftreten. In vielen Fällen eröffnet ein finanzielles Engagement den investierenden Unternehmen überdies die Möglichkeit, die Ausrichtung der entsprechenden Projekte im Sinne ihrer Partikularinteressen mitzugestal-

ten (etwa durch einen Sitz im *managing board*). Eine exemplarische Zusammenschau der Open Source Projekte, die auf der Plattform Open Hub in der ersten Jahreshälfte 2015 am populärsten waren sowie mehr als 2.000 Änderungen zwischen 5/2014 und 5/2015 aufweisen (Tab. 6), führt vor Augen, dass die meisten aktiven Projektgemeinschaften nicht ohne die finanzielle Unterstützung großer Unternehmen auskommen.

Tabelle 6: Populäre Open Source Projekte auf Open Hub

	Commits (5/14–5/15)	Assoziierte Organisation	Finanzierung (neben Entwicklungsarbeit)
Android (mobiles OS)	93.586		Google, Open Handset Alliance
KVM (Linux Virtual Machine) **Linux Kernel**	70.730 64.658	Linux Foundation	Spenden, Mitglieder (u.a. HP, Intel, IBM, NEC, Oracle, Samsung, Qualcomm)
Chromium (Browser)	68.268	Google	Google et al.
OpenStack (Cloud Computing)	61.941	OpenStack Foundation	Mitglieder (u.a. HP, IBM, Intel, Red Hat)
Mozilla Core (Bibliothek) **Mozilla Firefox** (Browser)	60.091 59.901	Mozilla Foundation	Royalties (2013: 95% Google), Spenden, Einblenden von Werbung
GNOME (Unix/Linux Desktop)	40.853	GNOME Foundation	Spenden (u.a. Google, IBM, Intel, FSF)
KDE (Arbeitsplatzumgebung)	36.220	KDE e.V.	Patronagen (u.a. Google, SUSE)
Debian (GNU/Linux OS)	26.374	Debian Project	Spenden, Partner (u.a. HP, 1&1)
LibreOffice (Bürosoftware)	20.157	Document Foundation	Spenden (u.a. Google, Red Hat, Intel)
WebKit (HTML Rendering)	14.867		Apple, Google, Amazon, Sony et al.
PHP (Skriptsprache)	9.385	PHP Group	Spenden, Ressourcen (u.a. Facebook)
GNU Compiler Collection	8.180	Free Soft. Foundation	Patronagen (u.a. Google, IBM), Spenden
Python (Programmiersprache)	5.208	Python Foundation	Sponsoring (u.a. Google, HP, Microsoft)
MySQL (relationale Datenbank)	4.917		Oracle
X.Org (Interface Protokoll)	4.487	X.Org Foundation	Spenden, Mitglieder (u.a. HP, IBM, AMD)
WordPress (CMS)	3.909	WP Foundation	Automattic, Veranstaltungen, Spenden
VLC Media Player	3.643	Video Lan NPO	Nutzerspenden, Partnerprogramme
Eclipse (IDE)	3.378	Eclipse Foundation	Mitglieder (u.a. Google, IBM, Oracle, SAP)
Apache HTTP Server **Apache Tomcat**	2.201 3.721	Apache Foundation	Sponsoring (u.a. Google, Microsoft, Facebook, Yahoo), Veranstaltungen
OpenSSL (Security)	938	OpenSSL Team	Seit 2014: Core Infrastructure Initiative
Arch Linux (OS)	355	—	Einzelspenden

Quelle: Open Hub, Jahresberichte (Stand: 6/2015). Ohne Tools/Frameworks.

Korporativ initiierte Entwicklungsprojekte stehen naturgemäß am eindeutigsten unter der finanziellen Ägide eines oder mehrerer Unternehmen. Das mobile Betriebssystem Android etwa wurde 2007 durch Google und die Open Handset Alliance lanciert, die 2015 aus 87 Firmen bestand, darunter vorrangig Mobilfunkbetreiber und Hardwarehersteller. Gängige Android-Distributionen funktionieren nur im Verbund mit proprietären Gerätetreibern und sind mit Google-eigenen Diensten verschränkt, was die Europäische Kommission 2015 zu einer kartellrechtlichen Prüfung anregte, für

die kooperierenden Firmen ob ihrer divergierenden Interessen aber unproblematisch sein dürfte. Insoweit lässt sich die Open Handset Alliance cum grano salis als ein klassisches *industry consortium* beschreiben (dazu: Werle 2000), das auf die Schaffung wechselseitig profitabler Standards ausgerichtet ist. Auch das durch Apple angestoßene WebKit Projekt dient primär diesem Zweck. MySQL hingegen ist ein Beispiel für ein Produkt mit dualem Lizenzsystem und direkt an Oracle angebunden.

Infrastrukturprojekte, deren Produkte übergreifenden Einsatz erfahren, werden ebenfalls vorwiegend durch Unternehmen getragen (vgl. Websites der Stiftungen): Neben der Allokation von Entwicklerressourcen werden die Projekte der Linux Foundation über die Jahresbeiträge korporativer Mitglieder finanziert, darunter 200 *silver members* (20.000+ US-Dollar), 14 *gold members* (100.000+ US-Dollar) und 8 *platinum members* (500.000+ US-Dollar), denen jeweils einer der 16 Sitze im Verwaltungsrat zugesprochen wird. Ein ähnliches Modell verfolgt die OpenStack Foundation, deren Hauptsponsoren (2015: HP, Canonical, AT&T, IBM, Intel, Rackspace, Red Hat, SUSE) gleichermaßen einen Sitz im Aufsichtsrat erhalten. Im Fall der Apache Foundation, die für 2013/2014 Zuwendungen von 1,1 Mio. US-Dollar ausweist, wird das *board of directors* hingegen komplett durch die Mitgliederbasis gewählt. Allerdings zeigt sich, dass dort mit Sam Ruby (IBM), David Nalley (Citrix) und Rich Bowen (Red Hat) ebenfalls Mitarbeiter größerer Anbieter sitzen. Zu den führenden Geldgebern gehörten 2015 u.a. Google, Facebook, Yahoo, Citrix, Cloudera und Microsoft.

Ein Sonderfall offener Infrastrukturstandards besteht in der Verschlüsselungssoftware OpenSSL, die in vielen kritischen Softwarearchitekturen (darunter z.B. auch Kreditkartensysteme) Einsatz findet. Trotz dieser Verbreitung wurde OpenSSL bis zur Formierung der Core Infrastructure Initiative (Kap. 3.2) im Wesentlichen durch einen Vollzeitprogrammierer sowie drei Freiwillige entwickelt und erhielt kaum finanzielle Unterstützung: „It's used by companies who make a lot of money, but almost none of the companies who use it contribute anything at all." (Ben Laurie in Perlroth 2014) Dieses Ungleichgewicht führte dazu, dass zwar immer neue Features integriert wurden, aber kaum Wartungsarbeiten stattfanden. Daraus resultierte 2012 eine gravierende Sicherheitslücke („Heartbleed"), die das Auslesen geschützter Daten auf fast allen IT-Geräten weltweit ermöglichte und erst 2014 entdeckt wurde (Stokel-Walker 2014).

Aber nicht nur Infrastrukturvorhaben und von Unternehmen initiierte Projekte, auch gesellschaftsethisch fundierte Communities wie KDE und GNOME (Desktopumgebungen) oder Debian (Linux-Distribution) rekurrieren auf Unternehmungszuwendungen: Die GNOME Foundation erhielt 2013 bei Gesamteinnahmen von 272.000 US-Dollar (ohne ‚Outreachy Program') 140.000 US-Dollar von Mitgliedern ihres Advisory Boards (u.a. Google, IBM, Intel, Red Hat); der hinter KDE stehende Verein nahm 2013 bei Einkünften von 190.292 Euro rund 77.000 Euro an korporativen Beiträgen und 79.000 Euro durch gesponserte Events ein; Debian unterhält Partnerschaften mit einer Vielzahl mittelgroßer Firmen und lässt sich Hardware und Hosting-Services fi-

nanzieren. Und auch die Free Software Foundation generierte 2013 über 80 Prozent ihres Umsatzes (1,2 Mio. US-Dollar) durch unternehmerische Zuwendungen.

Nicht in diese Trias einordnen lässt sich die Mozilla Foundation, die 2003 aus der mit Netscape/AOL assoziierten Mozilla Organization hervorgegangen ist (Kap. 2.4) und inzwischen über 1000 bezahlte Mitarbeiter beschäftigt (Hardy/Bilton 2014). Für 2013 wies die Mozilla Foundation (2014) Einnahmen von 314 Mio. US-Dollar aus (2012: 311 Mio. US-Dollar), davon 97 Prozent aus Lizenzeinnahmen für die Integration von Suchanbietern in die Firefox-Browserleiste, die bis 2013 hauptsächlich aus Vereinbarungen mit Google resultierten. Seit 2014 heißt der voreingestellte Suchdienst in den USA hingegen Yahoo. Ebenfalls seit 2014 schaltet Mozilla in Firefox *sponsored tiles* (Werbung). Mit Blick auf die Zahl fester Mitarbeiter, die finanziellen Kennzahlen und die Umsatzstrategien hat Mozilla nur noch wenig mit einem Open Source Vorhaben gemein, sondern lässt sich eher mit einem klassischen Softwareanbieter vergleichen.

Die meisten großen Open Source Projekte stehen heute insofern in einem finanziellen Wechselverhältnis mit führenden IT-Konzernen, die entlang ihrer Marktstrategien gezielt in spezifische Entwicklungsvorhaben investieren. Im Falle korporativ initiierter Projekte liegt diese Verschränkung auf der Hand; aber auch stiftungsgetragene Communities sprechen ihren Geldgebern Sitze in den *boards* ihrer Dachorganisationen zu, welche die Entwicklungsaktivitäten zwar zumeist nicht direkt steuern, aber die Infrastrukturen dafür verwalten und finanzielle Ressourcen verteilen. Kombiniert mit ihrem Involvement in die konkrete Code-Entwicklung sichern sich die jeweiligen Unternehmen so einen nicht zu unterschätzenden Einfluss auf relevante Entwicklungsvorhaben und tragen zugleich zu einer Erhöhung ihrer Planungssicherheit bei.

4 Spielarten quelloffener Softwareprojekte

Insgesamt lassen sich vor dem Hintergrund der steigenden sozioökonomischen Relevanz des IT-Sektors im Allgemeinen drei wesentliche Trends in der Entwicklung von Open Source Projekten und quelloffener Software herausstellen:

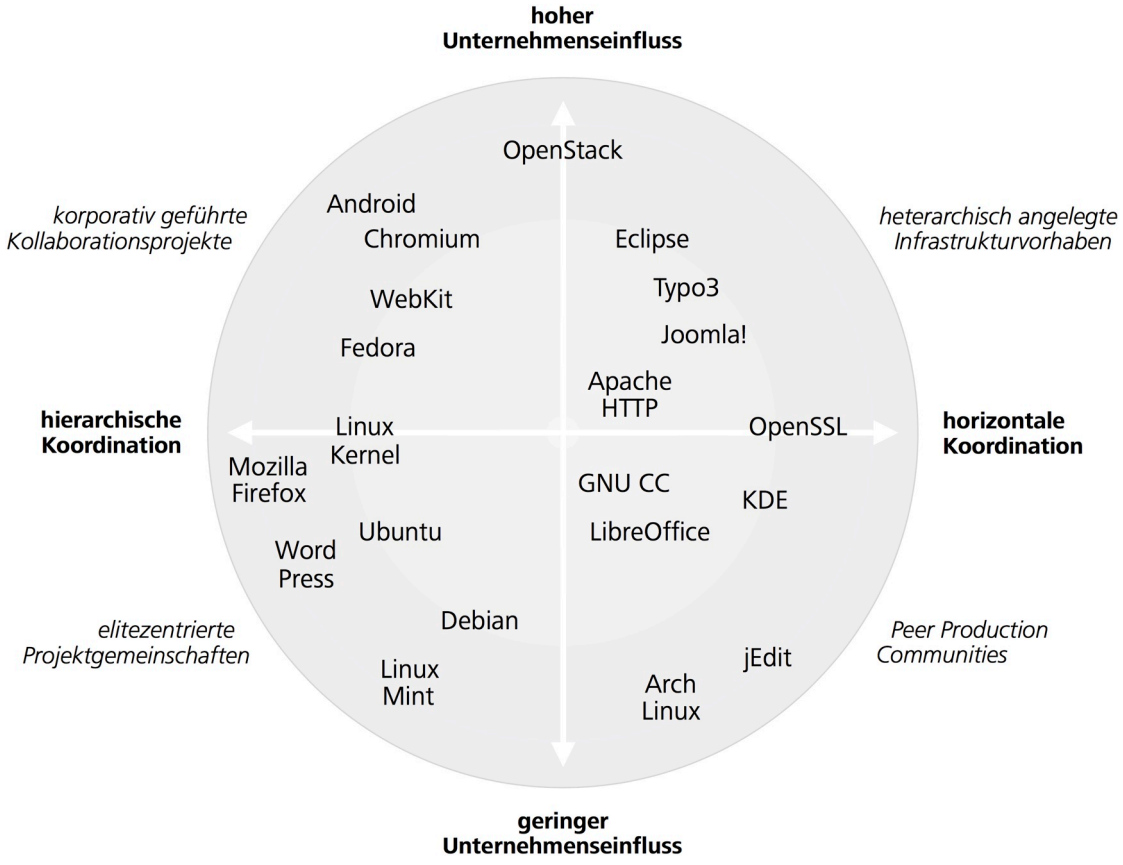
- Zum ersten haben sich ab Ende der 1980er Jahre sowohl *rechtlich belastbare Lizenzdefinitionen* für quelloffene Softwareprojekte etabliert, die deren Arbeitsergebnisse wirksam vor Einzelaneignung schützen, als auch übergreifend akzeptierte *Regeln, Werte und Arbeitskonventionen* herauskristallisiert, deren Ausdifferenzierung und Verbreitung durch die kommunikationserleichternden Eigenschaften der Onlinetechnologien erheblich befördert wurde.
- Zum zweiten sind Open Source Komponenten in den letzten 20 Jahren zu einem *integralen Bestandteil des Softwaremarktes* geworden und spielen heute sowohl für die basalen Infrastrukturen des Internets als auch in der Unternehmensinfor-

matik eine zentrale Rolle. Darüber hinaus wirken restriktiv und quelloffen lizenzierte Komponenten nicht nur im Zuge ihrer konkreten Indienstnahme ineinander, sondern werden in vielen Softwarepaketen bewusst miteinander kombiniert.

- Zum dritten lässt sich damit einhergehend eine zunehmende *Korporatisierung quelloffener Softwareprojekte* beobachten: Etablierte Unternehmen haben sich auf deren institutionelle Rahmenbedingungen eingestellt, nutzen entsprechende Kontexte zur Ergänzung ihrer proprietären Entwicklungsarbeit und zur Kooperation mit Mitbewerbern, protegieren für ihr Geschäft förderliche Projekte und haben neue Dienstleistungsfelder rund um Open Source erschlossen.

Im Horizont dieser Dynamiken hat sich eine große Bandbreite an sehr unterschiedlich ausgerichteten Vorhaben um Open Source Software herausgebildet: Am einen Ende des Spektrums finden sich Gemeinschaften, die nach wie vor Richard Stallmans gesellschaftsethischen Maximen verpflichtet sind, unabhängig von korporativen Interessen operieren und sich weitgehend an egalitären Organisationsprinzipien ausrichten; am anderen Ende lässt sich eine Vielzahl an Projekten identifizieren, die hierarchischen Koordinations- und Entwicklungsmodellen folgen sowie unter der unmittelbaren Kontrolle großer Technologieunternehmen stehen.

Abbildung 9: Idealtypische Varianten quelloffener Softwareprojekte



Entlang ihrer Koordinationsweisen sowie dem Grad ihrer Unternehmensnähe lassen sich derzeit vier typische Spielarten quelloffener Entwicklungsvorhaben voneinander abgrenzen (Abb. 9): (1) *Korporativ geführte Kollaborationsprojekte* für die Erarbeitung gemeinsamer Produkte, die durch ein leitendes Unternehmen gesteuert werden; (2) *heterarchisch angelegte Infrastrukturvorhaben*, die durch dezentralere Entscheidungsmuster geprägt sind, meist nicht durch kommerzielle Anbieter initiiert wurden, aber heute primär auf deren Leistungen basieren; (3) *elitezentrierte Projektgemeinschaften*, die sich durch einen moderateren Einfluss spezifischer Unternehmen auszeichnen, aber von eng definierten Kerngruppen geführt werden; und (4) eine relativ kleine Zahl an *Peer Production Communities* in Benklers (2002) Sinne, welche sich durch eine marktunabhängige Kollaboration unter Gleichberechtigten auszeichnen.

4.1 Korporativ geführte Kollaborationsprojekte

Ein Gutteil der marktrelevanten Open Source Projekte steht in einem direkten Verweisungszusammenhang mit großen IT-Konzernen, die in den letzten 15 Jahren zunehmend eigene Entwicklungsvorhaben angestoßen haben, anstatt ihre Aktivitäten auf existente Projektkontexte zu gründen (West/Bogers 2014; Riehle 2012). Angesichts der Dominanz von angestellten Entwicklern unter den Beiträgern steht dabei inzwischen weniger die Abschöpfung von „voluntary external work“ (Schaarschmidt et al. 2015: 100) im Vordergrund, als vielmehr die projektbezogene Kollaboration mit anderen industriellen Stakeholdern, um gemeinsam Produkte zu erarbeiten und entsprechende Synergieeffekte ohne die Ausgestaltung formaler Kooperationsbeziehungen zu nutzen. Im Folgenden werden mit Android, WebKit und Fedora drei Beispiele für unternehmensgeführte Open Source Projekte vorgestellt, bevor OpenStack als Mischform aus Infrastrukturvorhaben und korporativ angeleiteter Kollaborationsplattform beleuchtet wird. Diese Projekte gehören gemessen an ihrer Aktivität und dem qua Basic Constructive Cost Model geschätzten Arbeitsaufwand (Boehm 1981; Trendowicz/Jeffery 2014) zu den bis dato bedeutsamsten Open Source Vorhaben (Tab. 7).

Tabelle 7: Kennzahlen ausgewählter korporativ geführter Kollaborationsprojekte

	Geschätzter Aufwand 9/2015, Basic COCOMO*	**Commits** (5/2014–5/2015)	**Lizenz-modell**	**Strategische Kontrolle**
Android	4.000+ Personenjahre	81.119	permissive	Google / OH Alliance
Chromium	5.500+ Personenjahre	61.454	permissive	Google
WebKit	1.500+ Personenjahre	14.867	permissive	Apple
Fedora (Pack./Doc.)	160+ Personenjahre	45.232	gemischt	Red Hat / Council
OpenStack	780+ Personenjahre	62.370	permissive	Sponsoren / Komitee

*Quelle: Open Hub. * Basic Constructive Cost Model (organisch [a=2.4, b=1.05]).*

Android wurde ab 2003 als Betriebsumgebung für Mobile Devices durch ein Startup-Unternehmen entwickelt, das 2005 für ca. 50 Mio. US-Dollar von Google aufgekauft wurde. Bis 2007 akquirierte Google zwei Duzend korporative Partner, baute ein „secret patent portfolio" (Claburn 2007) auf und stellte Ende 2007 das Projekt sowie die Open Handset Alliance öffentlich vor. Die Entscheidung, Android als Open Source Vorhaben anzulegen, lässt sich zum einen darauf zurückführen, dass das System auf dem Linux Kernel und weiteren quelloffenen Komponenten fußt, die ohnehin nicht proprietarisierbar sind. Zum anderen flexibilisiert der Rückgriff auf freie Lizenzen die Zusammenarbeit zwischen den inzwischen knapp 90 beteiligten Unternehmen. Dabei ging es Google von vornherein weniger darum, mit Android selbst Umsatz zu generieren, als vielmehr den Zugriff auf seine Dienste zu erweitern, was sich auch in der soziotechnischen Verfasstheit des Projekts zeigt. Android-eigener Code steht unter permissiven Lizenzen, welche die Einbindung in kommerzielle Produkte erleichtern und Google gleichzeitig weitreichende Steuerungsmöglichkeiten einräumen, die sich beispielsweise im ‚Contributor Agreement' wie folgt widerspiegeln:

> „You hereby grant to the Project Leads and to recipients of software distributed by the Project Leads a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare derivative works of, publicly display, publicly perform, sublicense, and distribute Your Contributions and such derivative works." (Google 2015b)

Google ist als Projektleiter lediglich dazu verpflichtet, den Source Code finalisierter Versionen zu publizieren, und kann so im Verbund mit weiteren Rahmensetzungen wie der ‚Compatibility Definition' (Google 2015c) und den integrierten Programmierschnittstellen die technischen Spezifikationen abstecken, denen die Produktumsetzung folgt. Dieses Top-Down-Management wirkt einerseits einer Fragmentierung der Architekturen entgegen und erhöht die Erwartungssicherheit, was zu der raschen Verbreitung des mobilen Betriebssystems beigetragen hat. Andererseits wird Android durch diese zentralisierten Entscheidungsmuster aber zugleich zu dem bislang „most closed open source project" (Vision Mobile 2011). Auch *Chromium* als das zweite große von Google initiierte Open Source Projekt wird direkt durch das Unternehmen gesteuert und verfügt über keine vollständig einsehbaren Koordinationsstrukturen.

WebKit ist eine 2001 durch Apple angestoßene Abspaltung der von der KDE Community entwickelten KHTML Engine zur Darstellung von Webinhalten und die Basis für Apples Browser Safari in OS X und iOS sowie zahlreiche umgebungsspezifische Browser (z.B. Sony Playstation). Ähnlich wie Android steht WebKit gemessen an aktiven Entwicklern und Reviewern eindeutig unter der Ägide seines Gründungsunternehmens, das auch die ‚Contributors Meetings' ausrichtet, verfügt im Unterschied dazu aber nur über wenige prädefinierte Richtlinien und dokumentiert öffentlich, welche Entwickler aus welchen Firmen (u.a. Samsung, Google, Intel, Adobe) an welchen Arbeiten beteiligt sind (Teixeira/Lin 2014; Bitergia 2013). Hinter WebKit steht zudem kein festgeschriebenes Firmenkonsortium; die beteiligten Entwickler arbeiten vielmehr sachbezogen auf individueller Ebene zusammen an einer Softwarearchitektur,

welche in zahlreiche herstellerspezifische Produkte eingeht. „Maybe this is the reason why it can work, with such as deep involvement of fiercely competing companies: letting developers discuss without representing companies help to keep decisions technical and neutral.“ (Gonzalez-Barahona/Robles 2013: 178) Allerdings lassen sich auch im Falle von WebKit keine voll ausdefinierten Entscheidungswege oder formal bestimmten Steuerungsgruppen identifizieren; das Projekt richtet sich respektive an seinen erfahrensten Entwicklern aus, die in aller Regel bei Apple angestellt sind.

Von solchen konzernseitig initiierten Vorhaben hebt sich auf den ersten Blick die verbreitete Linux-Distribution *Fedora* ab, die 2002 im Rahmen eines studentischen Informatikprojektes entstanden ist und im Unterschied zu Android oder WebKit über ein gewähltes technisches Komitee verfügt, das Richtungsentscheidungen trifft. Auch Fedora ist jedoch eng mit einem auf seinem Feld marktführenden Unternehmen verknüpft, obgleich sich Red Hat gemessen an seinem Jahresumsatz (2014: 1,5 Mrd. US-Dollar) nicht mit Apple oder Google vergleichen lässt (Kap. 3.3): 2003 beschloss Red Hat, seine Linux-Distribution in eine kommerzielle Version und eine freie Variante aufzuspalten, die in das Fedora-Projekt überführt wurde. 2005 war Fedora kurzzeitig ein stiftungsgetragenes Vorhaben; aus steuerrechtlichen Gründen entschied sich Red Hat jedoch bereits 2006 dazu, die Fedora Foundation wieder aufzulösen. Seitdem gehört Fedora juristisch zu Red Hat und wird durch ein *board* bzw. seit 2014 durch ein konsensorientiertes *council* geführt, das sich zu zwei Dritteln aus gewählten Mitgliedern und zu einem Drittel aus Red Hat Mitarbeitern zusammensetzt, darunter auch der vetoberechtigte Projektleiter. Zudem speist sich die Teilnehmerbasis des Projekts neben Studierenden primär aus bei Red Hat angestellten Entwicklern und korporativen Nutzern von Red Hat Enterprise Linux. Finanziert wird Fedora fast vollständig durch das Unternehmen Red Hat, das zudem regelmäßig Fedora Events ausrichtet.

Die 2010 unter anderem durch die NASA angestoßene Entwicklung der Cloud Computing Architektur *OpenStack* hingegen wird durch kein Einzelunternehmen gesteuert, sondern durch die Beiträge von mittlerweile über 130 involvierten Firmen getragen, die fallweise für sie wesentliche Ressourcen in das Vorhaben investieren. Formal werden die Rahmenbedingungen des Projekts durch die 2012 gegründete OpenStack Foundation definiert; auf Arbeitsebene allerdings verteilt sich die Kontrolle der Entwicklungsarbeit neben einem gewählten Komitee auf die langfristig aktivsten industriellen Stakeholder, die als Platinsponsoren auch das Steuerungsboard der Stiftung beherrschen, darunter HP, IBM, Intel, AT&T und Rackspace sowie die Linux Distributoren Red Hat, Canonical und SUSE. Insofern lässt sich OpenStack als eine im Vergleich zu Android deutlich horizontaler organisierte *community of companies* charakterisieren, in der regelmäßig aktive Unternehmen indes eine privilegierte Rolle einnehmen. „This new kind of community [...] is clearly driven by corporate interests. Participating companies, which may be commercial competitors, have clear strategies towards the project [...].“ (Gonzalez-Barahona et al. 2013: 39)

In allen betrachteten Fällen setzen sich die Projektgemeinschaften vorrangig aus angestellten Entwicklern zusammen, die problemzentriert auf individueller Ebene oder als explizite Vertreter ihrer Unternehmen zusammenarbeiten. Eine solche korporative Kollaboration in Open Source Kontexten trägt zur Überwindung zweier *knowledge sharing dilemmas* (Cabrera/Cabrera 2002) bei: Zum einen verhindern quelloffene Lizenzen die Einzelaneignung des kollektiv erarbeiteten Codes; zum anderen stellen sie sich Trittbrettfahrern entgegen, die von den Resultaten kollektiven Engagements profitieren wollen, ohne eigene Ressourcen einzubringen, da es in Open Source Projekten stets nachvollziehbar bleibt, welche Firmen sich auf welche Elemente stützen und inwiefern sie an deren Entwicklung partizipieren (Henkel et al. 2014). Zudem liegt es in der Schöpfung von Softwareprodukten inzwischen ohnehin oft näher, statt einer kompletten Neuentwicklung auf gegebenen quelloffenen Architekturen aufzubauen. Eine Finanzierung von Open Source Projekten kann im Zweifel deutlich günstiger sein als die Lizenzierung proprietärer Lösungen oder die Verletzung von Schutzrechten.

Die konkrete Entwicklungsarbeit erfolgt in den diskutierten Entwicklungsvorhaben auf eigenen technischen Plattformen in Kombination mit Standardtools wie der Versionsverwaltung Git oder dem Fehlermeldungssystem Bugzilla, welche die ortsungebundene Zusammenarbeit vereinfachen. Korporativ geführte Kollaborationsprojekte haben gleichwohl nicht mehr viel mit dezentral strukturierten Communities gemein, sondern zeichnen sich durch prägnante Hierarchisierungen aus. Im Falle von Android, WebKit sowie Fedora liegt die strategische Kontrolle eindeutig bei den Einzelunternehmen Google, Apple und Red Hat; im Falle von OpenStack haben IT-Konzerne wie Hewlett-Packard, IBM oder Intel, die mit sehr vielen Programmierern in die Entwicklung involviert sind und über die finanziellen Mittel verfügen, um als zentrale Sponsoren aufzutreten, ebenfalls einen steuernden Einfluss auf das Projekt.

4.2 Heterarchisch angelegte Infrastrukturvorhaben

Von korporativ geführten Projekten heben sich graduell gewachsene Vorhaben ab, die auf basale Standards und Infrastrukturen fokussieren, eher horizontalen Organisationsmustern folgen und sich zwar ebenfalls auf die Leistungen etablierter Unternehmen stützen, aber nicht deren Kontrolle unterliegen. Sie werden oft durch Stiftungen getragen und weisen sich wandelnde Beteiligungsmuster auf (Wasserman 2013). In solchen heterarchisch angelegten Infrastrukturvorhaben geht es weniger um benennbare Produkte als um die Entwicklung übergreifend eingesetzter Werkzeuge – von integrierten Entwicklungsumgebungen wie Eclipse, über Content Management Systeme wie Typo3 oder Joomla bis hin zu Webserver-Software wie Apache HTTP und Verschlüsselungslösungen wie OpenSSL. Die meisten dieser Projekte existieren bereits seit über einem Jahrzehnt, zeichnen sich durch eine stabilisierte Code-Basis aus und weisen daher oft ein geringeres Aktivitätsniveau als jüngere Vorhaben auf (Tab. 8).

Tabelle 8: Kennzahlen ausgewählter Infrastrukturvorhaben

	Geschätzter Aufwand 9/2015, Basic COCOMO	**Commits** (5/2014–5/2015)	**Lizenz-modell**	**Strategische Kontrolle**
Eclipse*	1.800+ Personenjahre	3.515	weakly protective	Foundation Board
TYPO3	1.100+ Personenjahre	3.381	strongly protective	TYPO3 Association
Joomla	120+ Personenjahre	3.266	strongly protective	Open Source Matters
Apache HTTP	495+ Personenjahre	2.356	permissive	Foundation Board
OpenSSL	120+ Personenjahre	1.467	permissive	Core Team

*Quelle: Open Hub, eigene Recherchen. * Teilprojekte Platform/Web Tools Platform.*

Eclipse ist eine integrierte Entwicklungsumgebung für verbreitete Programmiersprachen, die auf der ab 1984 von IBM vertriebenen Plattform VisualAge fußt. Das 2001 unter freie Lizenz gestellte Projekt wurde zunächst durch ein Konsortium geführt, bis die Kontrolle 2004 an die Eclipse Foundation übergegeben wurde, die für 2014 rund 220 korporative Mitglieder und Einnahmen von 4,3 Mio. US-Dollar auswies, welche sich aus Mitgliedsbeiträgen speisen und primär der Bezahlung angestellter Entwickler dienen (Eclipse 2015). In ihrem achtzehnköpfigen Steuerungsboard sitzen neben vier gewählten Mitgliedern wechselnde Vertreter größerer Unternehmen (derzeit u.a. IBM, SAP, Oracle, Google, Red Hat), die Einfluss auf die Gesamtausrichtung des Projekts nehmen und seit 2015 über „targeted corporate contributions“ (Eclipse 2015b) auch spezifische Teilvorhaben fördern können. Trotz dieser Unternehmensnähe versteht sich Eclipse als offene meritokratische Gemeinschaft („the more you contribute the more responsibility you will earn“), die Wert auf transparente Entscheidungsmuster legt. Neben einsehbaren Sitzungsprotokollen und Mailinglisten lassen sich sämtliche Daten zu Programmaktualisierungen abrufen. Die konkrete Entwicklung erfolgt in eigenständigen Teams, die sich entlang gewählter Entscheidungsträger organisieren.

Auch die Content Management Systeme (CMS) *Typo3* und *Joomla* können auf (teil-) proprietäre Vorläufer zurückblicken und werden durch die Beiträge unternehmensaffiliierter Entwickler vorangetrieben: Typo3 wurde ab 1998 durch Kasper Skårhøj zunächst im Alleingang entwickelt und unter anderem aus pragmatischen Gründen („I couldn’t create the quality I wanted“) nach einer kommerziellen Phase unter freie Lizenz gestellt (Skårhøj in Phlow 2003). Joomla entstand 2005 als Abspaltung des ab 2000 durch die Firma Miro unter dualer Lizenz vertriebenen Mambo CMS, dessen Kernentwickler sich nach einem Konflikt mit Miro dafür entschieden, das Projekt eigenständig unter neuem Namen weiterzuführen. Zu diesem Zweck wurde das Non-Profit-Unternehmen Open Source Matters (Einnahmen 2014: 0,45 Mio. US-Dollar; primär aus Sponsorships) gegründet, das wie die 2004 als Verein konstituierte Typo3 Association (Einnahmen 2014: 0,85 Mio Euro; primär aus Mitgliedsbeiträgen) für

die Sicherung der Rahmenbedingungen zuständig ist, aber nicht in die technische Entwicklung eingreift (OSM 2015; T3A 2015). Während sich Typo3 als formal auch so gefasste „community of commercial actors“ charakterisieren lässt (Westenholz 2007: 11, 2012), arbeiten die Joomla Entwickler auf individueller Ebene zusammen. Sowohl Joomla als auch Typo3 erfahren vor allen Dingen durch kleinere und mittlere Unternehmen Unterstützung, die im Bereich Webdesign tätig sind. Beide Vorhaben sind heterarchisch entlang von Arbeitsgruppen organisiert, verfügen aber inzwischen über gewählte Führungsebenen für die übergreifende Projektkoordination.

Der *Apache HTTP Server* wurde 1994 von einer zunächst kleinen Gruppe an Programmierern entworfen, die mit den gegebenen Lösungen unzufrieden waren: „[...] in fact, the absence of a good commercial alternative was a powerful motivation for the launching of the project.“ (Lerner/Tirole 2002: 208) Aufgrund seiner permissiven Lizenz erfuhr der Apache HTTP Server schnelle Verbreitung und ist so früh zu einem zentralen Bestandteil des Softwaremarktes geworden (Greenstein/Nagle 2014). Zudem galt das Projekt rasch als Paradebeispiel für eine onlinebasierte selbstgesteuerte Kollaboration unter Freiwilligen (Benkler 2002). Bereits 1998 ging Apache freilich eine erste Partnerschaft mit IBM ein (McHugh 1998). Mittlerweile gehören zu der ca. 130 Personen starken Kerngruppe des Apache HTTP Projekts mehrheitlich Entwickler aus mittleren und großen IT-Firmen (z.B. IBM, HP, Fujitsu). Daneben genießen der HTTP Server und weitere Projekte über die Apache Software Foundation finanzielle Unterstützung durch marktführende Unternehmen wie Google, Facebook und Microsoft. Trotz dieses intensiven korporativen Engagements haben sich die Stiftung und ihre Projekte operative Unabhängigkeit bewahrt, was sich auch auf ihre frühzeitig ausgebildeten eigenständigen Koordinationsstrukturen zurückführen lässt: Während das Steuerungsboard der Stiftung durch ihre Mitglieder gewählt wird, die diesen Status einzig durch Qualifikation erlangen können, werden Positionen in der Entscheidungshierarchie des Apache HTTP Projekts leistungsbezogen vergeben – von einfachen Beiträgern über stimmberechtigte ‚committers‘ bis hin zum ‚management committee‘, welches strategische Entscheidungen trifft (Apache 2015).

Über solche ausdefinierten Koordinationsstrukturen verfügt die hinter der Verschlüsselungssoftware *OpenSSL* stehende Projektgemeinschaft nicht. OpenSSL galt lange trotz seines ubiquitären Einsatzes als Beispiel für ein basarartig strukturiertes Vorhaben, das ohne korporative Förderung operiert. Die 2014 entdeckte Sicherheitslücke „Heartbleed“ führte jedoch vor Augen, dass das Projekt seiner Marktrelevanz seit geraumer Zeit nicht mehr gewachsen war (Perlroth 2014): Seit 1999 wurde OpenSSL fast ausschließlich durch den einzigen angestellten Vollzeitmitarbeiter Stephen Henson und drei freiwillige Helfer aktualisiert, die ohne ausformulierte Regeln über Mailing-Listen kollaborieren. Erst seit 2014 wird OpenSSL über die Core Infrastructure Initiative in einem signifikanten Maße durch die Softwareindustrie unterstützt (Kap. 3.4) und befindet sich insoweit auf dem Weg zu einem korporativ geförderten Infra-

strukturprojekt. Gemessen an den auf der Plattform Open Hub aggregierten Aktivitätsdaten fußt das Vorhaben jedoch nach wie vor auf der Arbeit eines kleinen Kernteams: Zwischen 7/2014 und 7/2015 finden sich rund 20 kontinuierlich involvierte Entwickler in den Statistiken – das ist nicht viel für ein branchenzentrales Projekt.

Die skizzierten Infrastrukturprojekte zeichnen sich in mehrfacher Hinsicht durch ihre Verwobenheit mit korporativen Kontexten aus: Entweder sie basieren in ihrem Ursprung auf herstellerspezifischen Architekturen oder sie waren aus sich selbst heraus durch ein rasches Wachstum gekennzeichnet, da sie Lösungen für zuvor nicht probat adressierte Bereiche boten, und aus diesem Grund früh für Unternehmen interessant. Heute werden Eclipse, Joomla, Typo3 und Apache durch das Engagement großer IT-Firmen getragen; ihre Communities werden aber im Unterschied zu expliziten Kollaborationsprojekten nicht durch einen korporativen Kernzirkel angeleitet, sondern operieren unter dem Dach gemeinnütziger Organisationen und sind horizontaler angelegt. Gleichwohl haben sich mit wachsender Projektgröße stufenartige Führungsstrukturen herausgebildet, die der Qualitätssicherung dienen und die Gemeinschaft im Falle divergierender Partikularinteressen zusammenhalten. Die übergeordneten Funktionsträger werden zumeist leistungsbezogen designiert; allerdings können sich von Unternehmen für das Projekt freigestellte Entwickler in der Regel intensiver als Hobbyisten in die Community einbringen und so eher Entscheidungspositionen erlangen.

4.3 Elitezentrierte Projektgemeinschaften

Elitezentrierte Projektgemeinschaften zeichnen sich ähnlich wie viele Infrastrukturvorhaben durch ein organisches Wachstum aus und stehen nicht unter der Ägide eines etablierten Unternehmens. Vielmehr werden Projekte wie der Linux Kernel, Debian und Ubuntu, WordPress oder Mozilla Firefox durch einen eng definierten Führungszirkel angeleitet, an dessen Spitze oft ihr Gründer steht. Auch wenn die Zentralstellung von Einzelpersonen der Idee einer selbstgesteuerten Kollaboration unter Gleichberechtigten entgegensteht, könnten – so argumentiert zumindest Raymond (2000) – Projektleiter in Open Source Kontexten nicht despotisch agieren, sondern müssten stets das Wohl der Gemeinschaft im Blick haben, da deren Entwickler andernfalls auf der Basis der gegebenen quelloffenen Architekturen jederzeit neue Gemeinschaften gründen könnten. Benkler (2013: 225) vertritt zudem die Meinung, dass Plattformen wie die Versionsverwaltung Git einer „internal ossification of power“ entgegenwirken: „The technical infrastructure as it develops is supporting more distributed models and providing easier pathways to challenge oligarchy [...].“ Für die nachfolgend thematisierten Entwicklungsvorhaben lässt sich ein solcher technikinduzierter Einflussverlust ihres Führungspersonals allerdings kaum diagnostizieren (Tab. 9).

Der *Linux Kernel* ist in den letzten 20 Jahren zu der Grundlage zahlreicher Infrastrukturen und Produktlinien geworden – von Flachbildfernsehern bis hin zu Bankautoma-

ten. Im Gegensatz zu korporativ geführten Projekten ist das früh zum Aushängeschild für Open Source Software avancierte Vorhaben nicht durch ein Unternehmen lanciert worden; es wird heute aber vorrangig von angestellten Entwicklern getragen, darunter zuvorderst Mitarbeiter der Konzerne IBM, Intel und Samsung (Kap. 2.4). Stabilität erfährt das Projekt neben wenigen festgeschriebenen Richtlinien durch „a lieutenant system built around a chain of trust" (Kernel.Org 2015), an dessen Spitze der Projektgründer als „benevolent dictator" (Rivlin 2003) steht: „Linus Torvalds is the final arbiter of all changes accepted into the Linux kernel." (Kernel.Org 2015b) Am Ende des zitierten Dokumentationsartikels wird zum Beispiel schlicht auf einen Mailinglistenbeitrag Torvalds' verwiesen, in dem er einen Vorschlag zur Änderung der Betreffzeile für Korrekturen ablehnt und sein „canonical email format" definiert. Torvalds Managementstil wird als konsensorientiert, aber entscheidungsbereit beschrieben (Epstein 2015; Li et al. 2012). Der Source Code ist für alle Versionen zugänglich; technische Diskussionen sind über die Linux Kernel Mailing List (500 Beiträge/Tag) einsehbar.

Tabelle 9: Kennzahlen ausgewählter elitezentrierter Projektgemeinschaften

	Geschätzter Aufwand 9/2015, Basic COCOMO	**Commits** (5/2014–5/2015)	**Lizenz-modell**	**Strategische Kontrolle**
Linux Kernel	5.900+ Personenjahre	65.406	strongly protective	L. Torvalds
Ubuntu*	760+ Personenjahre	32.485	(strongly) protective	M. Shuttleworth
Linux Mint	140+ Personenjahre	1.439	(strongly) protective	C. Lefebvre / Team
Debian	24.800+ Personenjahre	21.076	(strongly) protective	Projektleiter
WordPress	100+ Personenjahre	3.621	strongly protective	M. Mullenweg
Mozilla Firefox	4.200+ Personenjahre	60.226	protective / permissive	M. Baker / Board

*Quelle: Open Hub; eigene Recherchen. * inklusive Ubuntu Touch.*

Ähnlich eindeutig sortiert sind die Einflussverhältnisse in den Projekten *Ubuntu* und *Mint*, deren Distributionen 2014 die meistgenutzten Linux-Varianten auf dem Desktop waren (DistroWatch 2015). Ubuntu wurde 2004 durch Mark Shuttleworth gegründet, der 1999 durch den Verkauf eines Startups zum Multimillionär geworden ist und als „self-appointed benevolent dictator for life (SABDFL) [...] a happily undemocratic role as sponsor of the project" spielt: „The SABDFL acts to provide clear leadership on difficult issues, and set the pace for the project." (Ubuntu 2015) Zudem ist das Shuttleworth unterstehende, 600 Mitarbeiter starke Unternehmen Canonical, das seinen Umsatz primär mit Dienstleistungen um Ubuntu macht, neben 500 Freiwilligen intensiv in das Projekt involviert. Auch das 2006 gegründete und von rund 20 Entwicklern getragene Linux Mint Projekt, hinter dem weder eine Firma noch eine Stiftung steht, richtet sich an den Entscheidungen seines Gründers Clement Lefebvre aus, der zwar rege Diskussionen schätzt, aber einen ebenso großen Wert auf

starke Führung legt: „The final decision comes from the top [...]. Strong leadership is important and benefits Linux Mint, [because] the decisions we take remain consistent and are coherent with our overall vision.“ (Lefebvre in Byfield 2013; Eitzen 2013)

Im Unterschied dazu steht im *Debian* Projekt ein demokratisch bestimmter Projektleiter an der Spitze, der befristet ähnliche Befugnisse wie Shuttleworth oder Torvalds hat. Debian Linux wurde 1993 durch den Studenten Ian Murdock initiiert, verfügte 2014 über ca. 1000 freiwillige Entwickler (Perrier 2014), gehört mit einem Marktanteil von 12 Prozent zu den verbreiteten Linux-Distributionen im Serverbereich und erhält neben Kleinzuwendungen über die NPO Software in the Public Interest (2015) primär Ressourcenspenden von mittelgroßen IT-Firmen. Seit 1997 verfügt Debian über einen Gesellschaftsvertrag, eine Verfassung und einen jährlich neu gewählten Projektleiter (Wahlbeteiligung 2015: 34 Prozent), der Mitglieder technischer Ausschüsse berufen, Ressourcen bzw. Verantwortungsbereiche verteilen und Ad-Hoc-Entscheidungen treffen kann (Debian 2015; Roeckx 2015). Seine Befugnisse sind das Resultat jahrelanger Aushandlungskontroversen (Coleman 2013; O'Mahony/Ferraro 2007). Unterhalb dieser Führungsebene lässt sich Debian als ‚bazaar of cathedrals' (Krafft 2010) beschreiben, in welchem delegierte Entwickler eigenständige Module betreuen. Neumitglieder müssen ein Prüfungsverfahren hinsichtlich ihrer Kompetenz und Verbundenheit mit den Projektleitlinien durchlaufen, darunter auch die Maxime, keine unfreie Software in das System einfließen zu lassen. Diese ideologische Fundierung gilt als ein Grund für die geringe Durchdringung des Projekts mit unternehmensaffiliierten Entwicklern.

Ausgeprägte institutionelle Strukturen lassen sich auch in der *WordPress* Community finden, die jedoch wiederum sehr gründerzentriert ist: WordPress wurde 2004 durch den Studenten Matt Mullenweg und den Webentwickler Mike Little veröffentlicht, ist heute das meistgenutzte Content Management System im Netz und wird durch ein zwanzigköpfiges Kernteam sowie eine meritokratisch strukturierte Projektgemeinschaft unter der alleinigen Leitung von Mullenweg aktualisiert. Anders als der Linux Kernel verfügt WordPress über mehrere ‚contributor handbooks' und ausdefinierte Richtlinien. Mullenweg ist zudem Vorsitzender der WordPress Foundation, Gründer der Investmentgesellschaft Audrey Capital und CEO des Unternehmens Automattic, das 2014 rund 350 *remote workers* beschäftigte und seinen Umsatz primär mit Enterprise Content Management macht. Angesichts ihrer zentralisierten Entscheidungsstrukturen vermutet Shane Snow (2014), dass alle genannten Firmen und Projekte im Falle eines Ausstiegs ihres Gründers zeitnah eingehen würden: „Despite the dictatorial nature of his management, it's Mullenweg's genuineness and vision that keeps WordPress's contributors contributing, Automattic's employees from churning.“

Das höchste Maß an Top-Down-Management unter den nicht unternehmensgeführten Projekten weist *Mozilla Firefox* auf. Im Gegensatz zu anderen Open Source Stiftungen nimmt die Mozilla Foundation über ihre Tochterfirma mit 1000 Angestellten direkten Einfluss auf die Projektarbeit: „The Mozilla Corporation [...] works with the

community to develop software that advances Mozilla's principles." (Mozilla 2015) Mit der Publikation des Firefox Browsers 2004 stellte Mozilla überdies auf ein „rather rigorously controlled model" (Stamelos 2014: 328) mit ausgeprägten Entscheidungshierarchien um – von „super-reviewers" über „release drivers" und „stewards" bis hin zu zwei „ultimate decision makers" (Mozilla 2015b). Eine dieser ultimativen Positionen nimmt seit 1998 die ehemalige Netscape-Managerin Mitchell Baker ein, die auch CEO der Mozilla Corporation und Vorsitzende der Mozilla Foundation ist; die zweite belegte bis 2014 der davor ebenfalls für Netscape tätige Brendan Eich. Führungsrollen in der Community werden leistungsorientiert vergeben; Entscheidungswege und Sitzungsprotokolle sind öffentlich einsehbar. Ein zentraler Orientierungspunkt ist das Mozilla Wiki, in dem auch kritische Punkte wie die Koordinationsprobleme zwischen angestellten Entwicklern und den nach eigenen Angaben mehr als 10.000 Freiwilligen thematisiert werden. In den *weekly updates* wurden zwischen 1/2015 und 7/2015 freilich über 95 neue Mitarbeiter, aber keine neuen Freiwilligen vorgestellt. Von den 314 Mio. US-Dollar Einnahmen im Jahr 2013 gab die Mozilla Foundation (2014) 65 Prozent für Entwicklung und 25 Prozent für Marketing und Verwaltung aus. Nicht nur hinsichtlich dieser Verteilungen, auch angesichts der hierarchischen Strukturen in der Stiftung und ihren Projekten folgt Mozilla eher klassischen Managementmustern.

Sowohl der Linux Kernel, WordPress und Mozilla als auch Mint und Ubuntu sind in ihren Entscheidungsabläufen auf ihre Gründer bzw. langjährigen Manager ausgerichtet, während die Debian Community jährlich einen Projektleiter wählt. Kritische Entschlüsse werden durch diese Führungspersonen im Verbund mit einem Kernzirkel gefasst, wobei alle genannten Projekte auch unterhalb dieser Steuerungselite über herausgehobene Entscheidungsträger verfügen, die vereinheitlichend tätig werden sowie Maßnahmen zur Qualitätssicherung ergreifen. All dies beschneidet die Spielräume der beteiligten Entwickler, bietet aber auch Orientierung und wirkt einer Fragmentierung der Projekte entgegen. Im Falle von Debian, Mozilla und WordPress sind deren tragende Richtlinien formal fixiert worden; im Falle von Mint und dem Linux Kernel haben sich entlang des Führungsstils ihrer Gründer hingegen lediglich „opaque governing norms" herausgebildet, die letztlich ihrer proklamierten Offenheit entgegenlaufen: „[...] without the law or a clear mechanism of accountability those injured by or excluded from peer production processes have very limited recourse." (Kreiss et al. 2011: 252) Insoweit sind es soziale Grundordnungen und nicht technische Strukturen per se, die allenfalls einer ‚internal ossification of power' entgegenwirken können.

4.4 Peer Production Communities

Von den bislang diskutierten Varianten lassen sich schließlich von korporativen Interessen weithin abgekoppelte Entwicklergemeinschaften unterscheiden, die sich durch „voluntaristic, nonstate, nonmarket action" bzw. „nonproprietary, [...] self-organized practices" auszeichnen (Benkler 2013: 213, 230), und deren Teilnehmer sich primär

aus intrinsischen Motiven einbringen: „Basically, people who participate in peer production communities love it. They feel passionate about their particular area of expertise [...].“ (Tapscott/Williams 2006: 70) Projektgruppen wie Arch oder jEdit entsprechen dieser Definition von Peer Production Communities passgenau, allerdings spielen ihre Produkte für den allgemeinen Markt kaum eine Rolle und die Zahl der involvierten Entwicklern ist überschaubar. Größere Vorhaben wie die GNU CC kommen hingegen inzwischen nicht mehr ohne basale Entscheidungshierarchien aus (Tab. 10).

Tabelle 10: Kennzahlen ausgewählter Peer Production Communities

	Geschätzter Aufwand 9/2015, Basic COCOMO	**Commits** (5/2014–5/2015)	**Lizenz-modell**	**Strategische Kontrolle**
GNU CC	2.200+ Personenjahre	8.382	strongly protective	Steuerungskomitee
KDE	7.000+ Personenjahre	36.220	(strongly) protective	Core Team
LibreOffice	240+ Personenjahre	21.384	weakly protective	Komitee / Board
jEdit	78+ Personenjahre	143	strongly protective	(B. Kautler) / Team
Arch Linux	19+ Personenjahre	323	(strongly) protective	(A. Griffin) / Team

Quelle: Open Hub; eigene Recherchen.

Arch wurde 2002 als GNU/Linux Distribution für Fortgeschrittene ins Leben gerufen und gilt unter Experten als beliebt, da es seinen Nutzern viele Freiheiten lässt. „Installing Arch Linux is a bit like building your own house. You have to dig the foundation, erect the walls, build the roofs, run the plumbing and electrical wiring around it [...].“ (Bhartiya 2015) Die Entwicklerbasis für das Kernsystem ist mit rund 30 aktiven freiwilligen Programmierern relativ klein; ihre Koordination erfolgt über Internet Relay Chat, Foren, Mailinglisten und ein öffentliches Wiki. Mit Aaron Griffin verfügt Arch über einen nominellen Projektleiter, der aber nicht die Position eines absoluten Entscheiders einnimmt. Entschlüsse werden auf der Basis konsensorientierter Diskussionen unter den Kernentwicklern gefasst; festgelegte Entscheidungswege und Kriterien für die Aufnahme in diesen inneren Kreis gibt es nicht. Im Arch Wiki (2015) findet sich lediglich eine Liste an Vorarbeiten, die Aspiranten dabei helfen können „to gain some ‚popularity‘ towards Arch’s developers“. Laut Griffin (in Cunningham 2010) folgt das Vorhaben ausschließlich intrinsischen Motiven: „We don’t make Arch so that we can win the most users, or get piles of cash. We make Arch because this is the OS we want to use.“ Arch erhält Kleinzuwendungen von Privatpersonen, aber keine nennenswerte Unterstützung durch die Softwareindustrie.

Auch *jEdit* verfügt über keine korporativen Sponsoren und richtet sich als javabasierter Texteditor für Programmierer an einen verhältnismäßig kleinen Nutzerkreis. Das Community Wiki bietet den Dreh- und Angelpunkt für die Projektkommunikation und listete Mitte 2015 über 300 Nutzer auf, von denen sich laut Open Hub (2015) al-

lerdings zwischen 7/2014 und 7/2015 lediglich 9 Personen in die Produktentwicklung eingebracht haben, darunter an vorderster Stelle Björn Kautler (alias ‚Vampire') und Alan Ezust, die gemessen an der Außenkommunikation der Gemeinschaft aus offenkundig nicht persönlich bekannten Entwicklern seit mehreren Jahren als informelle Projektmanager fungieren. Darüber hinaus verfügt die offene Entwicklergemeinschaft („You can ‚join' simply by subscribing to the [...] mailing lists") über keine fixierten Arbeitsregeln oder Rollenverteilungen; die Koordination findet hauptsächlich über eine projekteigene Webplattform inklusive Foren, Dateiverwaltung und Mailinglisten statt. jEdit wird oft als typisches Beispiel für ein „traditional OSS system" (Capiluppi et al. 2012: 197) in Benklers Sinne genannt; sowohl die Marktrelevanz des Texteditors als auch das Aktivitätsniveau in der Gemeinhaft hat zuletzt jedoch abgenommen.

Die seit 1996 entwickelte Linux-Arbeitsplatzumgebung *KDE* hingegen findet in Bildungseinrichtungen weltweit verbreiteten Einsatz und diente in den letzten Jahren als Ausgangsbasis für zahlreiche Projekte größerer Softwareunternehmen (z.B. WebKit). Dementsprechend beteiligen sich viele mittlere und größere Firmen an der Entwicklung der KDE Pakete, darunter Google, SUSE und Blue Systems, welche das Projekt über den gemeinnützigen KDE e.V. auch finanziell unterstützen (Kap. 3.4). Nichtsdestoweniger orientiert sich die Projektgemeinschaft nach wie vor an den originären Maximen freier Softwareentwicklung und bemüht sich um egalitäre Koordinationsweisen. 2014 bestand die KDE Community aus 500 aktiven Entwicklern, die sich über Mailinglisten, Foren und Repositorien entlang von Modulen selbstgesteuert organisieren. Anders als Debian verfügt KDE über keine fixierte Verfassung, keinen Projektmanager und auch der gewählte Vorstand des KDE Vereins greift nicht in die Softwareentwicklung ein (Alleyne 2011). Gleichwohl haben sich aufgrund der hohen Zahl an Entwicklern übergeordnete Managementstrukturen ausgebildet: 2008 wurde eine ‚Community Working Group' installiert, die als „mediator upon request" agiert; daneben entscheidet ein mehrere Dutzend Personen umfassendes ‚Core Team', das sich aus den verdientesten Entwicklern der Gemeinschaft zusammensetzt, öffentlich und konsensorientiert über die weitere strategische Ausrichtung des Projekts.

LibreOffice weist im Vergleich dazu explizitere Abstufungen in der Projektsteuerung auf. LibreOffice ist eine Abspaltung von OpenOffice.org, die 2010 durch die Kernentwickler der Community betrieben wurde, da diese mit der Projektpolitik des damals anleitenden Unternehmens Oracle unzufrieden waren. Mittlerweile ist LibreOffice die empfohlene Bürosoftware aller führenden Linux-Distributionen und verfügt über eine deutlich aktivere Projektgemeinschaft als OpenOffice, wobei knapp 40 Prozent der Änderungen zwischen 3/2014 und 3/2015 (22.134 Commits) von bei Red Hat angestellten Entwicklern getätigt worden sind (Corbet 2015). Die LibreOffice Gemeinschaft organisiert sich selbstständig entlang von Arbeitsgruppen; sie untersteht dabei aber unmittelbar den strategischen Setzungen und dem Qualitätsmanagement der mit dem Projekt gegründeten Document Foundation, die über ein gewähltes

board of directors, ein meritokratisch bestimmtes Steuerungskomitee und ein *advisory board* verfügt. Mitglieder dieses Beirats sind Organisationen, die einen jährlichen Beitrag von 5.000 bis 20.000 US-Dollar entrichten oder Mitarbeiter für die Projektentwicklung abstellen, darunter Google, Red Hat, SUSE, AMD, Intel und die Stadt München. Zudem erhält die Document Foundation (Einnahmen 2014: 0,8 Mio. Euro) finanzielle Unterstützung durch ein breites Portfolio an Softwareunternehmen.

Die *GNU Compiler Collection* (GCC), die seit 1987 als Teil des durch Richard Stallman initiierten GNU Projekts entwickelt wird, verfügt inzwischen ebenso über ausdefinierte Managementstrukturen. Die GCC übersetzt den Code höherer Programmiersprachen in Prozessor- bzw. Maschinensprache und findet in zahlreichen Produkten Einsatz, weshalb unternehmensaffiliierte Programmierer intensiv in das Vorhaben involviert sind. Aus diesem Grund wurde bereits 1998 ein ‚GCC steering committee' installiert – „with the intent of preventing any particular individual, group or organization from getting control over the project" (gcc.gnu.org/steering.html). Dieses konsensorientierte Komitee setzt sich neben Stallman aus 13 Mitgliedern zusammen, welche die verschiedenen Anspruchsgruppen des Vorhabens repräsentieren sollen. 9 Mitglieder dieser Führungsgruppe nehmen diese Rolle bereits seit 1998 ein; 10 Mitglieder arbeiten bei größeren Unternehmen wie IBM, Google oder Red Hat. Zudem verfügt GCC über einen Release Manager, einen fixierten Entwicklungsplan, der auf Diskussion in Mailinglisten basiert, ‚global reviewers' für das Gesamtprojekt sowie ‚maintainers' für die informell strukturierten Arbeitsmodule. Finanzielle Unterstützung erhält das Projekt über die Free Software Foundation, die 2013 ca. 1,2 Mio. US-Dollar an Spenden erhielt (u.a. von Google, IBM, HP) und 0,6 Mio. US-Dollar an Personalkosten auswies, die primär in die Anstellung von Entwicklern geflossen sind.

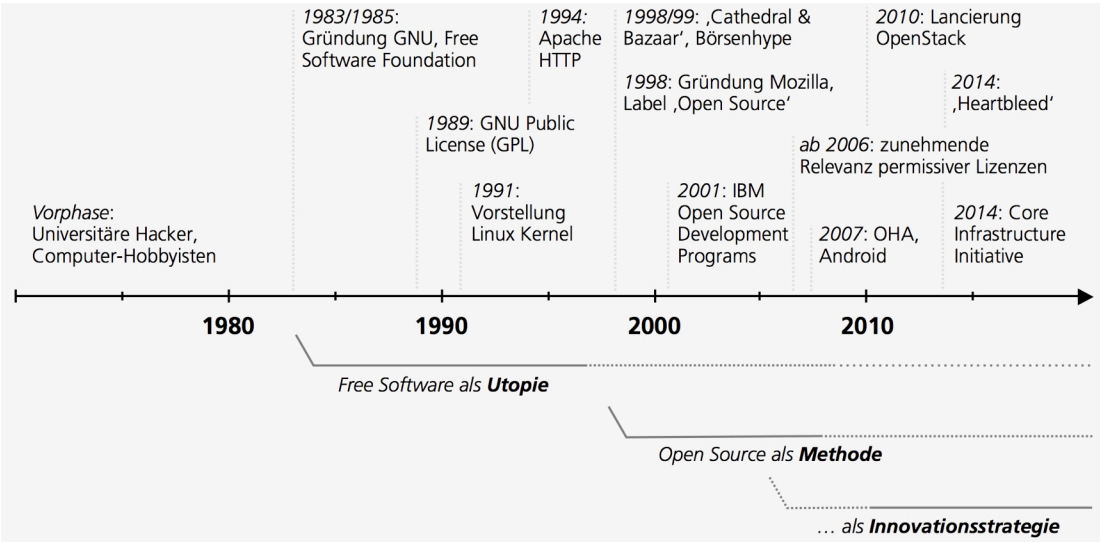
Auch in klassischen Peer Production Communities bilden sich insofern mit wachsender Größe hierarchische Entscheidungsstrukturen mit eindeutigen Machtasymmetrien und definierten Richtlinien heraus, die zur Kohärenz der Projekte beitragen und die übergreifende Koordination erleichtern. Unterhalb dieser formalen Managementstrukturen lässt sich im Unterschied zu elitezentrierten Gemeinschaften ein heterogeneres Bouquet an informellen Arbeitsgruppen ausmachen. Im Falle von GNU, KDE und LibreOffice bestimmt sich die Projektaktivität heute gleichwohl zu einem großen Teil durch die Beiträge angestellter Entwickler, die sich weniger aus individueller Passion engagieren, sondern da sie dafür bezahlt werden. Die Entwicklergemeinschaften Arch und jEdit richten ihre Produkte hingegen auf sehr spezifische Anspruchsgruppen aus, sind für korporative Stakeholder eher uninteressant, werden durch kleine Entwicklerteams getragen und konnten daher bis dato auf die Ausbildung ausgeprägter sozialer Strukturierungen verzichten. Sobald allerdings die Gemeinschaft wächst und sich ihre Interaktionen mit externen Akteuren intensivieren, werden augenscheinlich trotz aller technischen Effektivierungen ‚kathedralartige' Koordinationsstrukturen notwendig. Auch ein Basar kann sich nur bis zu einem gewissen Grad selbst organisieren.

5 Bilanz: Open Source als Utopie, Methode und Innovationsstrategie

Unter der Bezeichnung ‚Open Source Communities' wird heute ein breites Spektrum an sehr unterschiedlich ausgerichteten Projekten in der Softwareentwicklung zusammengefasst. Der überwiegende Teil dieser Vorhaben lässt sich freilich kaum mehr mit Eric S. Raymonds (1999: 21) Vorstellung eines selbstorganisierten „great babbling bazaar" oder Yochai Benklers (2002: 375) Idee einer „commons-based peer production" in Bezug bringen. Vielmehr haben die meisten marktrelevanten Projektgemeinschaften inzwischen prägnante, in aller Regel hierarchisch abgestufte Führungs- und Entscheidungsstrukturen ausgebildet und sind in einem signifikanten Maße von dem personellen wie finanziellen Involvement großer Unternehmen abhängig.

In der Retrospektive zeigt sich zudem, dass die quelloffene und proprietäre Entwicklung seit jeher ineinander verwoben sind: Erste Interessengemeinschaften von Computernutzern wurden in den 1950er Jahren durch Unternehmen mitbegründet; die ab den 1960er Jahren im akademischen Milieu entstandenen Hackergruppen verwendeten oft gespendetes Equipment; staatliche sowie privatwirtschaftliche Einrichtungen investierten zur gleichen Zeit bereits intensiv in universitäre Softwareprojekte; die Hobbyistenszene der 1970er Jahre fußte auf ebendiesen Vorarbeiten. Das gemeinsame Problem dieser meist in informellen Zusammenhängen entwickelten Architekturen bestand indes in ihrer mangelnden rechtlichen Absicherung: Die Ergebnisse der Zusammenarbeit wurden – wie auch in früheren Prozessen kollektiver Invention (Allen 1983) – als gemeinfreie Produkte veröffentlicht und waren nur unzureichend vor Einzelaneignung geschützt. Die eigentliche Geschichte der Open Source Softwareentwicklung beginnt vor diesem Hintergrund erst mit den Ende der 1980er Jahre definierten neuartigen Lizenzmodellen und lässt sich in drei Phasen einteilen (Abb. 10).

Abbildung 10: Phasen der Open Source Softwareentwicklung



Ab 1983 wurde ‚Free Software' durch Richard Stallman zunächst als gesellschaftsethisch fundierte Alternative zur proprietären Softwareherstellung und als *utopischer Gegenentwurf* zu kapitalistischen Wirtschaftsstrukturen formatiert. Die durch ihn gegründete Free Software Foundation erarbeitete in den Folgejahren die ersten rechtlich belastbaren ‚Copyleft'-Lizenzen, welche garantieren, dass auch Weiterentwicklungen freier Software einzig unter den gleichen Bedingungen verbreitet werden dürfen. Bereits in dieser Phase quelloffener Software als proklamierter Vorbote einer „high-tech gift economy" (Barbrook 1998) wurden allerdings vielversprechende Projekte wie der Linux Kernel durch einzelne Unternehmen wie IBM unterstützt.

Befördert durch den New Economy Hype der ausgehenden 1990er Jahre sowie die Vermarktungsbemühungen der 1998 gegründeten Open Source Initiative avancierte die quelloffene Softwareentwicklung mit der Jahrtausendwende zunehmend zu einer *allgemeinen Branchenmethode*. Kleinere und größere Technologieunternehmen investierten stetig intensiver in Open Source Projekte, um für ihre Belange förderliche Infrastrukturen und Standards zu protegieren, die Interoperabilität eigener Lösungen abzusichern oder ihre proprietären Programmpakete mit quelloffenen Komponenten zu ergänzen. Darüber hinaus bildete sich eine Reihe an Startup-Firmen heraus, die mit quelloffener wie kostenfreier Software im Verbund mit kostenpflichtigen Beratungs- und Supportdienstleistungen ein Geschäft aufbauen wollten.

Parallel dazu ist ‚Open Source' seit Mitte des letzten Jahrzehnts zu einem *festen Bestandteil der Innovationsstrategien* aller marktführenden Anbieter geworden, welche so ihre internen Forschungs- und Entwicklungsaktivitäten, die auf dem Markt für Informationstechnologien im Normalfall durch strikte Schließung geprägt sind, um lizenzrechtlich abgesicherte Kollaborationskontexte mit anderen Marktteilnehmern und kreative Spielflächen ergänzen. Apple etwa hat neben WebKit jüngst die Programmiersprache Swift unter freie Lizenz gestellt; Google hat sich bereits 2007 dazu entschlossen, Android als Open Source Projekt anzulegen. Diese Entscheidungen resultieren allerdings weniger aus ideeller Verbundenheit mit quelloffener Software, sondern vornehmlich aus ökonomischem Kalkül – zum Beispiel um die Attraktivität eigener Hardwareumgebungen zu erhöhen oder um neue Geschäftsfelder zu erschließen.

In den zurückliegenden 20 Jahren ist die Open Source Entwicklung auf diese Weise zu einem integralen Bestandteil der Softwareindustrie geworden; sie hat dabei aber ihre Formatierung als Gegenentwurf zur kommerziellen Herstellung weitgehend verloren. Derzeit lassen sich vier idealtypische Varianten von Open Source Projekten unterscheiden, die sich mit Blick auf ihre vorherrschenden Koordinationsmuster und dem Grad ihrer Nähe zu etablierten Unternehmen wie folgt voneinander abheben (Tab. 11):

- *Korporativ geführte Kollaborationsprojekte* wie Android oder WebKit sind durch Einzelunternehmen oder Unternehmensverbünde gegründet worden, eindeutig an deren Interessen orientiert und zeichnen sich durch entsprechend deutliche Hie-

rarchisierungen aus. Ihre Teilnehmerbasis speist sich vorrangig aus den Mitarbeiterkreisen der involvierten Firmen, die diese quelloffenen Projektzusammenhänge nutzen, um außerhalb formal gefasster Kooperationsbeziehungen mit anderen industriellen Stakeholdern häufig marktzentrale Produkte zu erarbeiten.

- *Heterarchisch angelegte Infrastrukturvorhaben* wie der Apache HTTP Server sind in der Regel graduell gewachsen und nicht korporativ initiiert worden, fußen heute in ihrer Operationsfähigkeit allerdings wesentlich auf dem finanziellen wie personellen Engagement mittlerer und großer Unternehmen. Sie verfügen über stabile eigenständige Koordinationsstrukturen, weisen wechselnde Beteiligungsmuster auf, sind eher horizontal strukturiert, verteilen Funktionsrollen oft meritokratisch und arbeiten unter der strategischen Führung einer assoziierten Dachorganisation.
- *Elitezentrierte Projektgemeinschaften* wie der Linux Kernel oder Ubuntu werden ebenfalls nicht durch einen gewerblichen Akteur kontrolliert; auch sie stützen sich aber inzwischen zu einem Gutteil auf die Beiträge unternehmensaffiliierter Entwickler. Ihre Koordination erfolgt entlang selektiver Entscheidungspyramiden, an deren Spitze meist entweder ein langfristig installiertes Führungsteam oder der Projektgründer als ‚benevolent dictator' steht, der mitunter in einem erheblichen Umfang an der Finanzierung des Projekts beteiligt ist.
- *Peer Production Communities* wie Arch Linux oder jEdit dienen ihrem eigenen Anspruch nach der marktunabhängigen, gleichberechtigten und selbstgesteuerten Kollaboration unter freiwilligen Entwicklern; sie bilden jedoch – wie sich am Beispiel der GNU Compiler Collection zeigen lässt – ab einer gewissen Marktrelevanz und Größe der Projektgemeinschaft ebenfalls herausgehobene Führungsstrukturen und basale hierarchische Abstufungen aus, welche die übergreifende Koordination erleichtern und zur Kohärenz des Vorhabens beitragen.

Tabelle 11: Open Source Softwareprojekte – idealtypische Ausprägungen

	Korporativ geführte Kollaborationsprojekte (z.B. Android, WebKit)	**Heterarchisch angelegte Infrastrukturvorhaben** (z.B. Apache HTTP Server)	**Elitezentrierte Projektgemeinschaften** (z.B. Linux Kernel, Ubuntu)	**Peer Production Communities** (z.B. Arch Linux)
Strategische Führung	Einzelunternehmen / Unternehmensgruppe	Vorstand der Dachstiftung / Dachorganisation	Projektgründer / langfristige Projektleitung	Komitee / Kernteam
Finanzierung	beteiligte Unternehmen	primär Zuwendungen von Unternehmen	korporative Spenden / gemischte Quellen	primär private Kleinspenden
Teilnehmer-basis	Mitarbeiter aus den beteiligten Unternehmen	angestellte Entwickler / Unternehmensvertreter	angestellte Entwickler / freiwillige Entwickler	freiwillige Entwickler
Arbeits-organisation	primär hierarchisch	horizontal / meritokratisch	hierarchisch / autokratisch	primär egalitär
Markt-relevanz	hoch	hoch	mittel / hoch	niedrig / mittel

Der gemeinsame Nenner aller betrachteten Entwicklungsvorhaben besteht in den dahinterliegenden quelloffenen Lizenzmodellen, die ihre Produkte wirksam vor direkter Proprietarisierung schützen und einen erwartungssicheren institutionellen Rahmen für die teleologische Zusammenarbeit zwischen Einzelentwicklern sowie Unternehmen bieten. Mit „Rebel Code“ (Moody 2002) oder „Coding Freedom“ (Coleman 2013) hat all dies gleichwohl nicht mehr viel gemein: Die Verschränkungen mit marktlichen Kontexten sind in vielen Fällen ausgeprägt; offener Quellcode mündet offenkundig nicht unmittelbar in transparenteren Koordinationsmustern als in anderen Arbeitszusammenhängen; trotz der technisch erweiterten Austauschmöglichkeiten bilden sich mit zunehmender Größe der Projektgemeinschaften regelmäßig hierarchisch gegliederte Entscheidungsstrukturen heraus. Zwar finden modulare, iterative und agile Methoden in allen diskutierten Vorhaben – wie seit geraumer Zeit auch in der unternehmensinternen Entwicklung (Cerri/Fuggetta 2007; Fuggetta 2003) – auf operativer Ebene verbreitete Anwendung; welchen langfristigen Schwerpunkten das Projekt folgt, definieren aber jeweils nur wenige Entscheidungsträger.

Entgegen dem Eindruck, „that the world of work is changing and that organizations (corporations, not-for-profits, universities) really don’t matter as much as they used to“ (Suddaby 2013: 1009), verlieren korporative Akteure in der Open Source Softwareprojekten überdies keineswegs an Bedeutung, sondern bleiben als deren Initiatoren, Financiers und Arbeitgeber der involvierten Programmierer prominent im Spiel. Privatwirtschaftliche Unternehmen, staatliche und wissenschaftliche Organisationen können ihre Ressourcen im Normalfall systematischer und auf lange Sicht verlässlicher als individuelle Akteure in die Projektgemeinschaften einbringen, tragen so zu einer Erhöhung der Planungssicherheit in den jeweiligen Kontexten bei und verfügen mithin oft über einen nicht zu unterschätzenden Einfluss auf deren Anlage und Orientierung. Ferner verfügen unabhängige Projekte, die nicht unter der direkten Ägide eines Unternehmens stehen, zumeist über assoziierte gemeinnützige Organisationen, die als adressierbare Dachidentitäten für externe Stakeholder dienen, die Ausrichtung des jeweiligen Vorhabens mitbestimmen und die Gemeinschaft bei Konflikten oder auseinanderstrebenden Partikularinteressen stabilisieren (Ahrne et al. 2014; Ahrne/Brunsson 2011; klassisch zu Organisationen: March/Simon 1958; Blau/Scott 1962).

Das nach wie vor einschlägige Narrativ von der quelloffenen Softwareentwicklung als revolutionäres Produktionsmodell, das auf freiwilliger sowie selbstgesteuerter Kollaboration beruht, eingespielten sozioökonomischen Koordinationsweisen überlegen ist und etablierte Unternehmen zu weitreichenden Anpassungen zwingt, lässt sich in seiner Radikalität insofern in den übergreifenden Entdifferenzierungsmythos der digitalen Moderne einordnen (Dickel/Schrape 2015): Ebenso wie im Falle anderer Spielarten kollektiven Handelns (z.B. in sozialen Bewegungen) oder kollektiver Produktion (z.B. in Wikis) zeigt sich auch mit Blick auf Open Source Communities, dass die Onlinetechnologien zwar die Kommunikation und Koordination erheblich effektivieren

und neuartige Möglichkeiten der Zusammenarbeit aufgeschlossen haben, aber grundlegende gesellschaftliche Strukturierungen und Rollendifferenzierungen dadurch keineswegs obsolet werden (Dolata/Schrape 2015). Die allermeisten der hier diskutierten Projektgemeinschaften haben sich mittlerweile denn auch zu äußerst komplexen sozialen Gebilden entwickelt, in denen sich klassische Organisationsprinzipien mit selektiv-hierarchischen Entscheidungsmustern und Einflussasymmetrien widerspiegeln.

Technische Infrastrukturen können die ortsunabhängige Kollaboration vereinfachen und auf operativer Ebene unterstützend wirken; die sozialen Ordnungsleistungen, die einen volatilen interessenbasierten Zusammenschluss in eine festgefügte Gemeinschaft mit der Fähigkeit zu strategischem Handeln überführen, können sie aber nicht ersetzen. Insoweit waren ab den 1980er Jahren nicht nur die neuen Formen der elektronischen Vernetzung, sondern ebenso die Kristallisation allgemein akzeptierter Werte bzw. Arbeitskonventionen sowie vor allem anderen die Definition rechtlich tragfähiger Lizenzmodelle für die Ausdifferenzierung und den anhaltenden Erfolg quelloffener Entwicklungsvorhaben von herausgehobener Bedeutung. ‚Copyleft'-Lizenzen und ihre Derivate sind nicht einfach nur eine vorübergehende „form of institutional jiujitsu" (Benkler 2002: 446) vor einer bestenfalls erhofften allgemeinen Dissolution geistiger Eigentumsrechte im Softwarebereich (Coleman 2013: 185–205), sondern nach wie vor das strukturelle Alleinstellungsmerkmal und die elementare Geschäftsgrundlage von Open Source Projekten, die inzwischen in einem unauflösbaren Verflechtungszusammenhang mit dem kommerziellen Markt stehen.

Literatur

Ahrne, Göran/Aspers, Patrik/Brunsson, Nils, 2014: The Organization of Markets. In: *Organization Studies*. Doi: 10.1177/0170840614544557 (ahead of print).

Ahrne, Göran/Brunsson, Nils, 2011: Organization Outside Organizations: The Significance of Partial Organization. In: *Organization* 18(1), 83–104.

Allen, Robert C., 1983: Collective Invention. In: *Journal of Economic Behavior and Organization* 4(1), 1–24.

Alleyne, Brian, 2011: Challenging Code: A Sociological Reading of the KDE Free Software Project. In: *Sociology* 45(3), 496–511.

Ante, Spencer, 2014: Red Hat Plays Hardball on OpenStack Software. In: *The Wall Street Journal* vom 13.5.2014. http://on.wsj.com/14qBpus (10/2015).

Apache Software Foundation, 2015: *Bylaws of The Apache Software Foundation*. http://www.apache.org/foundation/bylaws.html (10/2015).

Apache Software Foundation, 2015b: *Form 990. Fiscal Year 2013/2014*. http://www.apache.org/foundation/records/990-2013.pdf (10/2015).

Aponovich, David/Powers, Stephen/Kesler, Steven, 2014: *From Geek To Enterprise Chic. Open Source Digital Experience Tools Gain Traction*. Research Report. Cambridge: Forrester Inc.

Arch Wiki, 2015: *Getting Involved*. https://wiki.archlinux.org/index.php/Getting_involved (10/2015).

Arrow, Kenneth J., 1962: Economic Welfare and the Allocation of Resources for Invention. In: Nelson, Richard (Hg.): *The Rate and Direction of Inventive Activity. Economic and Social Factors*. Princeton: Princeton University Press, 609–626.

Baldwin, Carliss/Hippel, Eric von, 2011: Modeling a Paradigm Shift: From Producer Innovation to User and Open Collaborative Innovation. In: *Organization Science* 22(6), 1399–1417.

Barbrook, Richard, 1998: The High-Tech Gift Economy. In: *First Monday* 3(12). http://firstmonday.org/ojs/index.php/fm/article/view/631/552 (10/2015).

Bashe, Charles J./Johnson, Lyle R./Palmer, John H./Pugh, Emerson W., 1986: *IBM's Early Computers*. Cambridge: MIT Press.

Benkler, Yochai, 2002: Coase's Penguin, or, Linux and ‚The Nature of the Firm'. In: *Yale Law Journal* 112, 369–446.

Benkler, Yochai, 2004: Intellectual Property: Commons-based Strategies and the Problems of Patents. In: *Science* 305(5687), 1110–1011.

Benkler, Yochai, 2006: *The Wealth of Networks. How Social Production Transforms Markets and Freedom*. New Haven: Yale University Press.

Benkler, Yochai, 2013: Practical Anarchism, Peer Mutualism, Market Power, and the Fallible State. In: *Politics & Society* 41(2), 213–251.

Benkler, Yochai/Nissenbaum, Helen, 2006: Commons-based Peer Production and Virtue. In: *Journal of Political Philosophy* 14(4), 394–419.

Bergquist, Magnus/Ljungberg, Jan/Rolandsson, Bertil, 2012: *Justifying the Value of Open Source*. ECIS 2012 Proceedings. http://aisel.aisnet.org/ecis2012/122/ (10/2015).

Bezroukov, Nikolai, 1999: A Second Look at the Cathedral and the Bazaar. In: *First Monday* 4(12). http://firstmonday.org/article/view/708/618 (10/2015).

Bezroukov, Nikolai, 1999b: Open Source Software Development as a Special Type of Academic Research. In: *First Monday* 4(10). http://journals.uic.edu/ojs/index.php/fm/article/view/696/606 (10/2015).

Bhartiya, Swapnil, 2015: Five Reasons I Roll with Arch Linux, and Why You Should Too. In: *IT World* vom 18.5.2015. http://www.itworld.com/article/2898189 (10/2015).

Bitergia Inc., 2013: *Report on the Activity of Companies in the WebKit Project*. http://blog.bitergia.com/2013/02/06/report-on-the-activity-of-companies-in-the-webkit-project/ (10/2015).

Black Duck Software Inc./North Bridge Venture Partners, 2015: *The Future of Open Source 2014*. http://www.northbridge.com/2014-future-open-source-survey-results-0 (10/2015).

Blau, Peter M./Scott, Richard W., 1962: *Formal Organizations. A Comparative Approach*. San Francisco: Chandler.

Boehm, Barry, 1981: *Software Engineering Economics*. Englewood Cliffs: Prentice-Hall.

Brooks, Frederick, 1975: *The Mythical Man-Month*. Reading: Addison-Wesley.

Bulajewski, Mike, 2011: The Peer Production Illusion, Part I. In: *MrTeaCup* vom 19.11.2011. http://www.mrteacup.org/post/peer-production-illusion-part-1.html (10/2015).

Burton, Grad, 2002: A Personal Recollection: IBM's Unbundling of Software and Services. In: *IEEE Annals of the History of Computing* 24(1), 64–71.

Byfield, Bruce, 2013: What Makes for a Community Distribution? In*: Linux Magazine* vom 14.7.2013. http://www.linux-magazine.com/What-makes-for-a-community-distribution (6/2015).

Cabrera, Angel/Cabrera, Elizabeth F., 2002: Knowledge-Sharing Dilemmas. In: *Organization Studies* 23(5), 687–710.

Campbell-Kelly, Martin, 2003: *From Airline Reservations to Sonic the Hedgehog. A History of the Software Industry*. Boston: MIT Press.

Canonical Group Ltd., 2014: *Report and Financial Statements 2013*. http://de.scribd.com/doc/199373896/Canonical-Group-Limited-Annual-Accounts-2013 (10/2015).

Capek, Peter G./Frank, Steven P./Gerdt, Steve/Shields, David, 2005: A History of IBM's Open-Source Involvement and Strategy. In: *IBM Systems Journal* 44(2), 249–257.

Capiluppi, Andrea/Stol, Klaas-Jan/Boldyreff, Cornelia, 2013: Exploring the Role of Commercial Stakeholders in Open Source Software Evolution. In: Hammouda, Imed/Lundell, Björn/Mikkonen, Tommi/Scacchi, Walt (Hg.): *Open Source Systems*. Heidelberg: Springer, 178–200.

Carhart, Richard, 1953: *A Survey of the Current Status of the Electronic Reliability Problem*. Santa Monica: Rand Corporation.

Cerri, Davide/Fuggetta, Alfonso, 2007: Open Standards, Open Formats, and Open Source. In: *Journal of Systems and Software* 80(11), 1930–1937.

Ceruzzi, Paul E., 1998: *A History of Modern Computing*. Cambridge: MIT Press.

Claburn, Thomas, 2007: Google's Secret Patent Portfolio Predicts gPhone. In: *InformationWeek* vom 19.9.2007. http://www.informationweek.com/d/d-id/1059389 (10/2015).

Coleman, Gabriella E., 2013: *Coding Freedom. The Ethics and Aesthetics of Hacking*. Princeton: Princeton University Press.

Connell, Chuck, 2000: *Open Source Projects Manage Themselves? Dream On*. IBM/Lotus Developers Network. http://www.chc-3.com/pub/manage_themselves.htm (10/2015).

Corbet, Jonathan, 2015: Development Activity in LibreOffice and OpenOffice. In: *LWN.net* vom 25.3.2015. https://lwn.net/Articles/637735/ (10/2015).

Corbet, Jonathan/Kroah-Hartman, Greg/McPherson, Amanda, 2009–2015: *Linux Kernel Development Report*. San Francisco: The Linux Foundation.

Core Infrastructure Initiative, 2015: *Mission Statement*. http://www.linuxfoundation.org/programs/core-infrastructure-initiative (10/2015).

Coté, Michael/Governor, James/O'Grady, Stephen, 2007: *Open Source Strategies*. Analysis Paper. http://redmonk.com/public/goingopensource/Open-Source-Strategies.pdf (10/2015).

Cowan, Robin/Jonard, Nicolas, 2003: The Dynamics of Collective Invention. In: *Journal of Economic Behavior & Organization* 52(4), 513–532.

Cunningham, Jordan S., 2010: Interview: Arch Linux Team. In: *OS News* vom 11.1.2010. http://www.osnews.com/story/22692/Arch_Linux_Team (10/2015).

Dahlander, Linus/Magnusson, Mats, 2008: How do Firms Make Use of Open Source Communities? In: *Long Range Planning* 41(6), 629–649.

Debian Projekt, 2015: *Verfassung für das Debian-Projekt 1.5* (ratifiziert am 9.1.2015). https://www.debian.org/devel/constitution (10/2015).

Deshpande, Amit/Riehle, Dirk, 2008: The Total Growth of Open Source. In: Russo, Barbara et al. (Hg.): *Open Source Development, Communities and Quality*. New York: Springer, 197–209.

Dickel, Sascha/Schrape, Jan-Felix, 2015: Dezentralisierung, Demokratisierung, Emanzipation. Zur Architektur des digitalen Technikutopismus. In: *Leviathan* 43(3), 442–463.

DistroWatch, 2015: *DistroWatch Page Hit Ranking*. http://distrowatch.com (10/2015).

Dolata, Ulrich, 2015: Volatile Monopole. Konzentration, Konkurrenz und Innovationsstrategien der Internetkonzerne. In: *Berliner Journal für Soziologie* 24(4), 505–529.

Dolata, Ulrich/Schrape, Jan-Felix, 2015: Masses, Crowds, Communities, Movements: Collective Action in the Internet Age. In: *Social Movement Studies*. Doi: 10.1080/14742837.2015.1055722 (ahead of print).

Driscoll, Kevin, 2015: Professional Work for Nothing: Software Commercialization and ‚An Open Letter to Hobbyists'. In: *Information & Culture* 50(2), 257–283.

Driver, Mark, 2013: *Open Source is Everywhere in Modern IT*. Präsentation, POSSCON 2013. Videodokument. https://www.youtube.com/watch?v=cVWcegd_ce (10/2015).

Driver, Mark, 2014: *Within the Enterprise, Open Source Must Coexist in a Hybrid IT Portfolio*. Gartner Inc. Research Report vom 23.8.2014. Stamford: Gartner.

Eclipse Foundation, 2015: *2015 Annual Eclipse Community Report*. https://eclipse.org/org/foundation/reports/annual_report.php (10/2015).

Eclipse Foundation, 2015b: *Eclipse Development Process 2014*. https://eclipse.org/projects/dev_process/ (10/2015).

Eitzen, Christopher von, 2013: *Q&A: Clement Lefebvre*. http://www.networkworld.com/article/2170903/software/q-a--clement-lefebvre--the-man-behind-linux-mint.html (10/2015).

Epstein, Richard A., 2015: Political Economy of Crowdsourcing: Markets for Labor, Rewards, and Securities. In: *The University of Chicago Law Review* 82, 35–52.

Europäische Kommission, 2015: *Kommission leitet förmliche Untersuchung zum mobilen Betriebssystem Android gegen Google ein*. European Commission MEMO 15/4782.

Fisher, Franklin M./McKie, James W./Mancke, Richard B., 1983: *IBM and the US Data Processing Industry: An Economic History*. Santa Barbara: Praeger.

Fitzgerald, Brian, 2006: The Transformation of Open Source Software. In: *MIS Quarterly* 30(3), 587–598.

Forrester Research, 2013: *Market Update: Office 2013 and Productivity Suite Alternatives*. Market Report. Cambridge: Forrester Inc.

Forrester Research, 2014: *Enterprise And SMB Software Survey, North America and Europe*. Market Report. Cambridge: Forrester Inc.

Fuggetta, Alfonso, 2003: Open Source Software–An Evaluation. In: *Journal of Systems and Software* 66(1), 77–90.

Free Software Foundation, 1986: *GNU's Bulletin* 1(1). https://www.gnu.org/bulletins/bull1.txt (10/2015).

Free Software Foundation, 1989: *GNU General Public License (GPL), Version 1.0*. http://www.gnu.org/licenses/old-licenses/gpl-1.0.html (10/2015).

Free Software Foundation, 2013: *Form 990*. http://static.fsf.org/nosvn/Form990_FY2013.pdf (10/2015).

Gartner Inc., 2015: *Market Share: All Software Markets, Worldwide, 2014*. Stamford: Gartner Inc.

Gates, Bill, 1976: An Open Letter To Hobbyists. In: *Computer Notes* 1(9), 3.

Gelsi, Steve, 1999: VA Linux Rockets 698%. In: *CBS Marketwatch* vom 10.12.1999. http://www.cbsnews.com/news/va-linux-rockets-698/ (10/2015).

Germain, Jack M., 2014: Adobe's Open Source Tightrope Walk. In: *Linux Insider* vom 19.4.2014. http://www.linuxinsider.com/story/80329.html (10/2015).

Ghosh, Rishab Aiyer et al., 2006: *Economic Impact of Open Source Software on Innovation and the Competitiveness of the Information and Communication Technologies (ICT) Sector in the EU*. Maastricht: UNU-MERIT.

Gillen, Al, 2013: *Open Source Software. The Foundation for Tomorrow's Infrastructure*. IDC Präsentation, Nagios World Conference 2013.

GNOME Foundation, 2013: *GNOME Annual Report 2013*. https://www.gnome.org//wp-content/uploads/2014/09/GNOME-Annual-Report-2013.pdf (10/2015).

Gonsalves, Antone, 2013: IBM Makes OpenStack the Cloud Platform to Beat. In: *ReadWrite* 5.3.2013. http://readwrite.com/2013/03/05/ibm-makes-openstack-the-cloud-platform-to-beat (10/2015).

Gonzalez-Barahona, Jesus M./Izquierdo-Cortazar, Daniel/Maffulli, Stefano, 2013: Understanding How Companies Interact with Free Software Communities. In: *IEEE Software* 30(5), 38–45.

Gonzalez-Barahona, Jesus M./Robles, Gregorio, 2013: Trends in Free, Libre, Open Source Software Communities: From Volunteers to Companies. In: *it–Information Technology* 55(5), 173–180.

Google Inc., 2010–2015: *Annual Report. Form 10-K*. https://investor.google.com (10/2015).

Google Inc., 2015b: *Android Corporate Contributor Agreement*. http://source.android.com/source/cla-corporate.html (10/2015).

Google Inc., 2015c: *Android Compatibility Definition*. http://static.googleusercontent.com/media/source.android.com/de//compatibility/android-cdd.pdf (10/2015).

Greenstein, Shane/Nagle, Frank, 2014: Digital Dark Matter and the Economic Contribution of Apache. In: *Research Policy* 43, 623–631.

Gulley, Ned/Lakhani, Karim, 2010: *The Determinants of Individual Performance and Collective Value in Private-collective Software Innovation*. Harvard Business School Technology & Operations Management Unit Working Paper 10/065.

Hammond, Jeffrey, 2014: *Open Source by the Numbers*. Präsentation, All Things Open Conference 2014. Videodokument. https://www.youtube.com/watch?v=GTFM39h5m5g (10/2015).

Hardy, Quentin/Bilton, Nick, 2014: Personality and Change Inflamed Mozilla Crisis. In: *New York Times* vom 4.4.2014, B1.

Henkel, Joachim/Schöberl, Simone/Alexy, Oliver, 2014: The Emergence of Openness: How and Why Firms Adopt Selective Revealing in Open Innovation. In: *Research Policy* 43(5), 879–890.

Herstatt, Cornelius/Ehls, Daniel, 2015: *Open Source Innovation: Phenomenon, Participant Behaviour, Business Implications*. New York: Routledge.

Hogan, Mike, 2009: The Corporate Closing of Open Source. In: *Planet MySQL* vom 22.5.2009. http://planet.mysql.com/entry/?id=19610 (10/2015).

Holtgrewe, Ursula/Werle, Raymund, 2001: De-Commodifying Software? Open Source Software between Business Strategy and Social Movement. In: *Science Studies* 14(2), 43–65.

Hortonworks Inc., 2015: *Financial Results for Fiscal Year 2014*. http://de.hortonworks.com/press-releases/hortonworks-reports-financial-results-fourth-quarter-fiscal-year-2014/ (10/2015).

IBM Corp., 1959: *IBM 650 Model 4 Announcement*. Press Release. http://www-03.ibm.com/ibm/history/exhibits/650/650_pr4.html (10/2015).

IBM Corp., 2013: *IBM Commits $1 Billion to Fuel Linux and Open Source Innovation on Power Systems*. Press Release. http://www-03.ibm.com/press/us/en/pressrelease/41926.wss (10/2015).

IBM Corp., 2015: *2014 Annual Report*. http://www.ibm.com/annualreport/2014/bin/assets/IBM-Annual-Report-2014.pdf (10/2015).

IBM Corp., 2015b: *Open Source and Standards*. http://www-03.ibm.com/linux/ossstds/ (10/2015).

International Data Corporation, 2012: *Worldwide Software Forecast*. http://s3.amazonaws.com/zanran_storage/www.bloomberg.com/ContentPages/2566616115.pdf (10/2015).

International Data Corporation, 2014: *Worldwide Server Market Revenues Decline -4.4% in Q4*. Press Release. http://www.idc.com/getdoc.jsp?containerId=prUS24704714 (10/2015).

Jaeger, Till, 2010: Enforcement of the GNU GPL in Germany and Europe. In: *Journal of Intellectual Property, Information Technology and E-Commerce Law* 1/2010, 34–39.

Jive Software Inc., 2014: *Form 10-K*. http://investors.jivesoftware.com (10/2015).

Johnson, Luanne, 2002: A View from the Sixties: How the Software Industry Began. In: Akera, Atsushi/Nebeker, Fredrik (Hg.): *From 0 to 1. An Authoritative History of Modern Computing*. New York: Oxford University Press, 101–109.

Juliussen, Karen/Juliussen, Egil, 1990: *The Computer Industry Almanac 1991*. New York: Brady.

KDE e.V., 2013: *Community Report 27*. https://ev.kde.org/reports (10/2015).

Kernel.Org Documentation, 2015: *How the Development Process Works*. https://www.kernel.org/doc/Documentation/development-process/2.Process (10/2015).

Kernel.Org Documentation, 2015b: *How to Get Your Change Into the Linux Kernel*. https://www.kernel.org/doc/Documentation/SubmittingPatches (10/2015).

Kolassa, Carsten/Riehle, Dirk/Riemer, Philip/Schmidt, Michael, 2014: Paid vs. Volunteer Work in Open Source. In: *Proceedings 47th Hawaii Int. Conference on System Sciences*, 3286–3295.

Kostakis, Vasilis/Papachristou, Marios, 2014: Commons-based Peer Production and Digital Fabrication. In: *Telematics and Informatics* 31(3), 434–443.

Krafft, Martin F., 2010: *A Delphi Study of the Influences on Innovation Adoption and Process Evolution in a Large Open Source Project: The Case of Debian*. Dissertation. Limerick: University of Limerick, Department of Computer Science & Information Systems.

Kreiss, Daniel/Finn, Megan/Turner, Fred, 2011: The Limits of Peer Production: Some Reminders from Max Weber for the Network Society. In: *New Media & Society* 13(2), 243–259.

Lakhani, Karim R./Hippel, Eric von, 2003: How Open Source Software Works. In: *Research Policy* 32(6), 923–943.

Leonard, Andrew, 2001: Life, Liberty and the Pursuit of Free Software. In: *Salon* vom 16.2.2001. http://www.salon.com/2001/02/15/unamerican/ (10/2015).

Lerner, Joshua, 2012: *The Architecture of Innovation*. Boston: Harvard Business Press.

Lerner, Joshua/Schankerman, Mark, 2010: *The Comingled Code. Open Source and Economic Development*. Cambridge: MIT Press.

Lerner, Joshua/Tirole, Jean, 2002: Some Simple Economics of Open Source. In: *Journal of Industrial Economics* 50(2), 197–234.

Lerner, Joshua/Tirole, Jean, 2005: The Scope of Open Source Licensing. In: *Journal of Law, Economics & Organization* 21(1), 20–56.

Lessig, Lawrence, 1999: Open Code and Open Societies: Values of Internet Governance. In: *Chicago Kent Law Review* 74, 1405–1420.

Levy, Steven, 1984: *Hackers: Heroes of the Computer Revolution*. Garden City: Anchor Press.

Lewine, Peter, 2014: Why There Will Never Be Another Red Hat: The Economics Of Open Source. In: *Techcrunch* vom 13.2.2014. http://on.tcrn.ch/l/pjXf (10/2015).

Li, Yan/Tan, Chuan/Teo, Hock-Hai, 2012: Leadership Characteristics and Developers' Motivation in Open Source Software Development. In: *Information & Management* 49(5), 257–267.

Lowood, Henry, 2009: Videogames in Computer Space: The Complex History of Pong. In: *IEEE Annals of the History of Computing* 31(3), 5–19.

March, James/Simon, Herbert, 1958: *Organizations*. Cambridge: Blackwell.

McAllister, Neil, 2013: Microsoft no longer a Top Kernel Contributor. In: *The Register* 16.9.2013. http://www.theregister.co.uk/2013/09/16/linux_foundation_kernel_report_2013/ (10/2015).

McGaw, Judith A., 1987: *Most Wonderful Machine: Mechanization and Social Change in Berkshire Paper Making 1801–1885*. Princeton: Princeton University Press.

McHugh, Josh, 1998: For the Love of Hacking. In: *Forbes* vom 10.8.1998. http://www.forbes.com/forbes/1998/0810/6203094a.html (10/2015).

Menell, Peter S., 2002: Envisioning Copyright Law's Digital Future. In: *New York Law School Review* 46, 63–199.

Meyer, Peter B., 2003: *Episodes of Collective Invention*. BLS Working Paper 368. Washington: U.S. Bureau of Labor Statistics.

Microsoft Corp., 2015: *2014 Annual Report*. http://www.microsoft.com/investor/reports/ (10/2015).

Microsoft Corp., 2015b: *Openness*. http://www.microsoft.com/en-us/openness/index.aspx (10/2015).

Moddy, Glyn, 2002: *Rebel Code. The Inside Story of Linux and the Open Source Revolution*. New York: Basic Books.

Mowery, David C., 1999: The Computer Software Industry. In: Ders./Nelson, Richard W. (Hg.): *Sources of Industrial Leadership*. Cambridge: Cambridge University Press, 133–168.

Mowery, David C./Langlois, Richard N., 1996: Spinning off and Spinning on(?): The Federal Government Role in the Development of the US Computer Software Industry. In: *Research Policy* 25(6), 947–966.

Mozilla Foundation, 2014: *Independent Auditors' Report and Consolidated Financial Statements*. https://static.mozilla.com/moco/en-US/pdf/Mozilla_Audited_Financials_2013.pdf (10/2015).

Mozilla Foundation, 2015: *Mozilla Organizations*. https://www.mozilla.org/en-US/about/ (10/2015).

Mozilla Foundation, 2015b: *Mozilla Roles and Leadership*. https://www.mozilla.org/en-US/about/governance/roles/ (10/2015).

Mulligan, Catherine, 2013: *The Communications Industries in the Era of Convergence*. London: Routledge.

NetApplications Inc., 2015: *Desktop Operating System Market Share*. http://www.netmarketshare.com/operating-system-market-share.aspx (10/2015).

Netcraft Ltd., 2015: *September 2015 Web Server Survey*. http://news.netcraft.com/archives/2015/09/16/september-2015-web-server-survey.html (10/2015).

Netscape Communications Corp., 1998: *Netscape Announces mozilla*.org. Press Release 23.2.1998.

Nuvolari, Alessandro, 2004: Collective Invention during the British Industrial Revolution: The Case of the Cornish Pumping Engine. In: *Cambridge Journal of Economics* 28(3), 347–363.

O'Mahony, Siobhán, 2003: Guarding the Commons. How Community Managed Software Projects Protect their Work. In: *Research Policy* 32(7), 1179–1198.

O'Mahony, Siobhán, 2005: Non-profit Foundations and their Role in Community-firm Software Collaboration. In: Feller, Joseph/Fitzgerald, Brian/Hissam, Scott A./Lakhani, Karim R. (Hg.): *Perspectives on Free and Open Source Software*. Cambridge: MIT Press, 23–46.

O'Mahony, Siobhán/Ferraro, Fabrizio, 2007: The Emergence of Governance in an Open Source Community. In: *Academy of Management Journal* 50(5), 1079–1106.

Open Hub, 2015: *Project Summaries and Contributors Listings*. https://www.openhub.net/ (10/2015).

Open Source Matters Inc., 2015: *Budget vs. Actuals – FY 2014*. http://opensourcematters.org/images/financials/2014/Budget_vs_Actuals__FY2014-12.pdf (10/2015).

Osterloh, Margit/Rota, Sandra, 2007: Open Source Software Development: Just another Case of Collective Invention? In: *Research Policy* 36(2), 157–171.

Perens, Bruce, 1999: The Open Source Definition. In: DiBona, Chris/Ockman, Sam/Stone, Mark (Hg.): *Opensources. Voices from the Open Source Revolution*. Sebastopol: O'Reilly, 171–188.

Perlroth, Nicole, 2014: Heartbleed Highlights a Contradiction in the Web. In: *The New York Times* vom 18.4.2014. http://nyti.ms/1hb6uBd (10/2015).

Perrier, Christian, 2014: Developers per Country. In: *Bubulle's Weblog* vom 29.7.2014. http://www.perrier.eu.org/weblog/2014/07/29#devel-countries-201308 (10/2015).

Phister, Montgomery, 1976: *Data Processing. Technology and Economics*. Bedford: Santa Monica.

Phlow Magazine, 2003: Typo3 und Kasper Skårhø. Interview. In: *Phlow* vom 29.4.2003. http://phlow.net/mag/interview_portrait/typo3_und_kasper_skrh.php (6/2015).

PriceWaterhouseCoopers AG, 2014: *PwC Global 100 Software Leaders*. London: PWC.

Raymond, Eric S., 1998: The Revenge of the Hackers. In: DiBona, Chris/Ockman, Sam/Stone, Mark (Hg.): *Opensources. Voices from the Open Source Revolution*. Sebastopol: O'Reilly, 207–220.

Raymond, Eric S., 1998b: *Goodbye, „free software"; Hello, „open source"*. Announcement vom 22.11.1998. ftp://ftp.lab.unb.br/pub/computing/museum/esr/open-source.html (10/2015).

Raymond, Eric S., 1999: *The Cathedral and the Bazaar. Musings on Linux and Open Source by an Accidental Revolutionary*. Sebastopol: O'Reilly.

Raymond, Eric S., 2000: Project Structures and Ownership. In: *Homesteading the Noosphere* (Version 3.0). http://catb.org/~esr/writings/homesteading/homesteading/ar01s16.html (10/2015).

Reimer, Jeremy, 2012: From Altair to iPad: 35 Years of Personal Computer Market Share. In: *Ars Technica* vom 14.8.2012. http://bit.ly/OwzHKm (10/2015).

Riehle, Dirk, 2012: The Single-Vendor Commercial Open Source Business Model. In: *Information Systems and E-Business Management* 10(1), 5–17.

Rifkin, Jeremy, 2014: *The Zero Marginal Cost Society: The Internet of Things, the Collaborative Commons, and the Eclipse of Capitalism*. New York: Palgrave Macmillan.

Rivlin, Gary, 2003: Leader of the Free World: How Linus Torvalds Became Benevolent Dictator of Planet Linux. In: *Wired* 11/2003, 157.

Roeckx, Kurt, 2015: *Debian Project Leader Election 2015 Results*. https://lists.debian.org/debian-devel-announce/2015/04/msg00003.html (10/2015).

Romer, Paul, 1990: Endogenous Technological Change. In: *Journal of Pol. Economy* 98(5), 71–102.

Sands, Rich, 2012: Measuring Project Activity. In: *Back Duck Open Hub Blog* vom 10.4.2012. http://blog.openhub.net/2012/04/measuring-project-activity/ (10/2015).

Sayadian, Helga, 1990: *The Information Technology Industry Data Book 1960–2000*. Washington: Computer and Business Equipment Manufacturers Association.

Schaarschmidt, Mario/Walsh, Gianfranco/Kortzfleisch Harald F., 2015: How Do Firms Influence Open Source Software Communities? In: *Information and Organization* 25(2), 99–114.

Schwarz, Michael/Takhteyev, Yuri, 2010: Half a Century of Public Software Institutions. In: *Journal of Public Economic Theory* 12(4), 609–639.

Singer, Harold, 1976: An Open Letter to Ed Roberts. In: *Micro-8 Computer User Newsletter* 2(4), 1.

Snow, Shane, 2014: How Matt's Machine Works. In: *Fast Company* vom 11.9.2014. http://www.fastcompany.com/3035463/how-matts-machine-works (10/2015).

Software in the Public Interest Inc., 2015: *2015 Annual Report*. http://www.spi-inc.org/corporate/annual-reports/2015.pdf (10/2015).

Spencer, Jennifer W., 2003: Firms' Knowledge-sharing Strategies in the Global Innovation System. Evidence from the Flat Panel Display Industry. In: *Strategic Management Journal* 24(3), 217–233.

Spies, Rüdiger, 2010: SAP und Open Source. In: *IT-Business* vom 14.12.2010. http://www.it-business.de/partnerzones/idc-research-zone/viewpoints/articles/ 296134/ (10/2015).

Spreeuwenberg, Kimberley/Poell, Thomas, 2012: Android and the Political Economy of the Mobile Internet. In: *First Monday* 17(7). http://dx.doi.org/10.5210/fm.v17i7.4050 (10/2015).

Stallman, Richard, 1983: *New UNIX Implementation*. http://bit.ly/1DSDoXW (10/2015).

Stallman, Richard, 1985: GNU Manifesto. In: *Dr. Dobb's Journal of Software Tools* 10(3): 30.

Stallman, Richard, 2002: *Free Software, Free Society*. Boston: GNU Press.

Stamelos, Ionnais, 2014: Management and Coordination of Free/Open Source Projects. In: Ruhe, Günther/Wohlin, Claes (Hg.): *Software Projekt Management in a Changing World*. New York: Springer, 321–341.

StatCounter Inc., 2015: *Global Stats*. http://gs.statcounter.com/ (10/2015).

Steinmueller, William E., 1995: The US Software Industry. An Analysis and Interpretive History. In: Mowery, David C. (Hg.): *The International Computer Software Industry. A Comparative Study of Industry Evolution and Structure*. New York: Oxford University Press, 15–52.

Stiller, Ase, 2011: The Open Source Trials: Hanging in the Legal Balance of Copyright and Copyleft. In: *Vision Mobile Blog* vom 15.3.2011. http://www.visionmobile.com/blog/2011/03/the-open-source-trials-hanging-in-the-legal-balance-of-copyright-and-copyleft/ (10/2015).

Stokel-Walker, Chris, 2014: The Internet Is Being Protected By Two Guys Named Steve. In: *Buzzfeed* vom 25.4.2014. http://www.buzzfeed.com/chrisstokelwalker/the-internet-is-being-protected-by-two-guys-named-st#.ooP0j2q2z (10/2015).

Suddaby, Roy, 2013: Book Review: The Janus Face of Commercial Open Source Software Communities. In: *Organization Studies* 34(7), 1009–1011.

Tapscott, Don/Williams, Anthony D., 2006: *Wikinomics. How Mass Collaboration Changes Everything*. New York: Portfolio.

Teixeira, Jose/Lin, Tingting, 2014: Collaboration in the Open-Source Arena: The WebKit Case. In: *Proceedings of the 52. ACM Conference on Computers and People*. New York: ACM, 121–129.

The Document Foundation, 2015: *2014 Annual Report*. https://wiki.documentfoundation.org/File:TDF2014AnnualReport.pdf (10/2015).

Torvalds, Linus, 1998: LINUX Manifesto. Interview. In: *BOOT Magazine* 7-8/1998, 32–37.

Torvalds, Linus, 2002: Re: [PATCH] Remove Bitkeeper Documentation from Linux Tree. In: *Linux Kernel Mailinglist* vom 20.4.2002. http://lwn.net/2002/0425/a/ideology-sucks.php3 (10/2015).

Trendowicz, Adam/Jeffery, Ross, 2014: Constructive Cost Model—COCOMO. In: Dies.: *Software Project Effort Estimation*. Heidelberg/New York: Springer, 277–293.

Tukey, John W., 1958: The Teaching of Concrete Mathematics. In: *American Mathematical Monthly* 65(1), 1–9.

Typo3 Association, 2015: *Financial Statements for the Year 2014*. http://typo3.org/fileadmin/t3org/images/FM-content/association/financial/ Financial_Statements_2014.pdf (10/2015).

Ubuntu Project, 2015: *Governance*. http://www.ubuntu.com/about/about-ubuntu (10/2015).

United Nations Conference of Trade and Development, 2012: *Information Economy Report 2012*. New York/Genf: United Nations.

Vision Mobile Ltd., 2011*: Open Governance Index*. London: Vision Mobile.

W3techs Surveys, 2015: *Technologies Overview*. http://w3techs.com/technologies (10/2015).

Warren, Jim C., 1976: Correspondence. In: *SIGPLAN Notices* 11(7), 1–2.

Wasserman, Anthony, 2013: Community and Commercial Strategies in Open Source Software. In: *Information Technology* 55(5), 181–188.

Watson, Richard T./Boudreau, Marie-Claude/York, Paul T./Greiner, Martina E./Wynn, Donald, 2008: The Business of Open Source. In: *Communications of the ACM* 51(4), 41–46.

Weber, Steven, 2005: *The Success of Open Source*. Cambridge: Harvard University Press.

Werle, Raymund, 2000: *Institutional Aspects of Standardization. Jurisdictional Conflicts and the Choice of Standardization Organizations*. MPIfG Discussion Paper 2000/1. Köln: MPIfG.

West, Joel/Bogers, Marcel, 2014: Leveraging External Sources of Innovation. A Review of Research on Open Innovation. In: *Journal of Product Innovation Management* 31(4), 814–831.

West, Joel/Dedrick, Jason, 2001: Open Source Standardization: The Rise of Linux in the Network Era. In: *Knowledge, Technology & Policy* 14(2), 88–112.

West, Joel/Salter, Ammon/Vanhaverbeke, Wim/Chesbrough, Henry, 2014: Open Innovation: The Next Decade. In: *Research Policy* 43(5), 805–811.

Westenholz, Ann, 2007: *Cannibalizing Your Own Business? Commercializing the Open Source Software TYPO3*. TII Working Paper Series, Volume 20. Edmonton: University of Alberta.

Westenholz, Ann (Hg.), 2012: *The Janus Face of Commercial Open Source Software Communities*. Copenhagen: Copenhagen Business School Press.

Weitere Publikationen

Stuttgarter Beiträge zur Organisations- und Innovationsforschung

Radig, Ann-Kathrin, 2015: *Der Wandel des deutschen Videoverleihmarktes durch Digitalisierung und Internet*. SOI Discussion Paper 2015-01.

Dolata, Ulrich, 2014: *Märkte und Macht der Internetkonzerne. Konzentration – Konkurrenz – Innovationsstrategien*. SOI Discussion Paper 2014-04.

Kungl, Gregor, 2014: *The Incumbent German Power Companies in a Changing Environment. A Comparison of E.ON, RWE, EnBW and Vattenfall from 1998 to* 2013. SOI Discussion Paper 2014-3.

Dolata, Ulrich/Schrape, Jan-Felix, 2014: *Masses, Crowds, Communities, Movements. Collective Formations in the Digital Age*. SOI Discussion Paper 2014-2.

Neukirch, Mario, 2014: *Konflikte um den Ausbau der Stromnetze. Status und Entwicklung heterogener Protestkonstellationen*. SOI Discussion Paper 2014-1.

Dolata, Ulrich/Schrape, Jan-Felix, 2013: *Zwischen Individuum und Organisation. Neue kollektive Akteure und Handlungskonstellationen im Internet*. SOI Discussion Paper 2013-2.

Kosche, Robert, 2013: *Kollektive Identitäten in Industrial Cultural Districts*. SOI Discussion Paper 2013-1.

Fuchs, Gerhard/Hinderer, Nele/Kungl, Gregor/Neukirch, Mario, 2012: *Adaptive Capacities, Path Creation and Variants of Sectoral Change*. SOI Discussion Paper 2012-2.

Fuchs, Gerhard/Wassermann, Sandra, 2012: *Organising a Market. Photovoltaics in Germany*. SOI Discussion Paper 2012-1.

Werle, Raymund, 2011: *Institutional Analysis of Technical Innovation. A Review*. SOI Discussion Paper 2011-04.

Dolata, Ulrich, 2011: *Radical Change as Gradual Transformation. Characteristics and Variants of Socio-technical Transitions*. SOI Discussion Paper 2011-03.

Dolata, Ulrich, 2011: *The Music Industry and the Internet*. SOI Discussion Paper 2011-02.

Schrape, Jan-Felix, 2011: *Der Wandel des Buchhandels durch Digitalisierung und Internet*. SOI Discussion Paper 2011-01.

Monographien und Herausgeberschaften

Dolata, Ulrich, 2013: *The Transformative Capacity of New Technologies. A Theory of Sociotechnical Change*. London: Routledge.

Dolata, Ulrich/Schrape, Jan-Felix (Hg.), 2013: *Internet, Mobile Devices und die Transformation der Medien. Radikaler Wandel als schrittweise Rekonfiguration*. Berlin: Edition Sigma.

Dolata, Ulrich, 2011: *Wandel durch Technik. Eine Theorie soziotechnischer Transformation*. Frankfurt/New York: Campus.

Fuchs, Gerhard/Shapira, Philip (Eds.), 2014: *Rethinking Regional Innovation. Path Dependency or Regional Breakthrough*. Springer: Shanghai (chinese edition).

Schrape, Jan-Felix, 2015: *Kommunikation und Partizipation im Social Web. Eine Übersicht*. Studienbrief der FernUniversität in Hagen.

Schrape, Jan-Felix, 2012: *Wiederkehrende Erwartungen. Prognosen, Visionen und Mythen um neue Medien seit 1970*. Boizenburg: VWH.

Schrape, Jan–Felix, 2011: *Gutenberg-Galaxis Reloaded?* Boizenburg: VWH.

Schrape, Jan–Felix, 2010: *Neue Demokratie im Netz? Eine Kritik an den Visionen der Informationsgesellschaft*. Bielefeld: Transcript.

Beiträge in Zeitschriften und Sammelbänden

Dolata, Ulrich/Schrape, Jan-Felix, 2015: Masses, Crowds, Communities, Movements: Collective Action in the Internet Age. In: *Social Movement Studies*. DOI: 10.1080/14742837.2015.1055722.

Dolata, Ulrich, 2015: Forschung und Entwicklung in der Wirtschaft: Konzentration, Kooperation und Internationalisierung. In: Simon, Dagmar/Knie, Andreas/Hornbostel, Stefan (Hg.): *Handbuch Wissenschaftspolitik*. Wiesbaden: Springer VS. DOI: 10.1007/978-3-658-05677-3_35-1.

Dolata, Ulrich, 2015: Volatile Monopole. Konzentration, Konkurrenz und Innovationsstrategien der Internetkonzerne. In: *Berliner Journal für Soziologie* 24(4), 505–529.

Dolata, Ulrich/Schrape, Jan-Felix, 2014: App Economy: Demokratisierung des Software-Marktes? In: *Technikfolgenabschätzung – Theorie und Praxis* 23(2), 76–80.

Dolata, Ulrich/Schrape, Jan-Felix, 2014: Kollektives Handeln im Internet. Eine akteurtheoretische Fundierung. In: *Berliner Journal für Soziologie* 24(1), 5–30.

Dolata, Ulrich/Schrape, Jan-Felix, 2014: Markt und Macht in der App-Economy. In: *Blätter für deutsche und internationale Politik* 4/2014, 31–34.

Dolata, Ulrich/Schrape, Jan-Felix, 2013: Medien in Transformation. Radikaler Wandel als schrittweise Rekonfiguration. In: Dies. (Hg.): *Internet, Mobile Devices und die Transformation der Medien*. Berlin: Edition Sigma, 9–37.

Dolata, Ulrich, 2012: Radikaler Wandel als graduelle Transformation. In: Michael Decker, Armin Grunwald, Martin Knapp (Hg.): *Der Systemblick auf Innovation*. Berlin: Edition Sigma, 95–106.

Dolata, Ulrich, 2011: Soziotechnischer Wandel als graduelle Transformation. In: *Berliner Journal für Soziologie* 21(2), 265–294.

Dolata, Ulrich, 2011: Google vs. Facebook: Der Kampf um das Internet. In: *Blätter für deutsche und internationale Politik* 9/2011, 26–29.

Fuchs, Gerhard, 2015: Building the Agenda for Carbon Dioxide Capture and Storage: Limits of EU-Activism. In: Tosun, Jale/Biesenbender, Sophie/Schulze, Kai (Eds.): *Energy Policy Making in the EU*. London: Springer, 205–223.

Fuchs, Gerhard/Hinderer, Nele, 2014: Situative Governance and Energy Transitions in a Spatial Context: Case Studies from Germany. In: *Energy, Sustainability and Society* 4:16.

Fuchs, Gerhard, 2014: Die Rolle lokaler Initiativen bei der Transformation des deutschen Energiesystems. In: *GAIA* 23(2), 135–136.

Fuchs, Gerhard, 2014: The Governance of Innovations in the Energy Sector: Between Adaptation and Exploration. In: *Science & Technology Studies* 27(1), 34–53.

Fuchs, Gerhard, 2014: Innovationen im Energiesektor als strategische Handlungsfelder. Die Governance von Anpassung und Erneuerung. In: Löw, Martina (Hg.): *Vielfalt und Zusammenhalt. Verhandlungen des 36. DGS Kongresses*. Frankfurt (Main): Campus, 675–691.

Fuchs, Gerhard/Wassermann, Sandra, 2012: From Niche to Mass Markets in High Technology: The Case of Photovoltaics in Germany. In: Johannes Bauer et al. (Eds.): *Innovation Policy and Governance in High-Tech Industries*. Heidelberg/Berlin: Springer, 219–244.

Kungl, Gregor, 2015: Stewards or Sticklers for Change? Incumbent Energy Providers and the Politics of the German Energy Transition. In: *Energy Research & Social Science* 8, 13–23.

Neukirch, Mario, 2015: Mehr Netz mit weniger Kohle? Zwei ökologische Perspektiven auf ‚Korridor D'. In: *politische ökologie* 141, 132–135.

Neukirch, Mario, 2013: Ausbau der Stromnetze – Konflikte und Perspektiven der deutschen Energiewende. In: *GAIA* 22(2), 138–139.

Neukirch, Mario, 2013: Offshore-Windkraft als Plan B der Energiekonzerne? In: *Berliner Debatte Initial* 24(1), 125–136.

Neukirch, Mario, 2012: Grüner Netzausbau für schmutzigen Strom? In: *Blätter für deutsche und internationale Politik* 6/2012, 25–28.

Neukirch, Mario, 2012: Internationale Nutzung der Windkraft zur Stromproduktion. In: Ehrhardt, Henrik/ Kroll, Thomas (Hg.): *Energie in der modernen Gesellschaft*. Göttingen: Vandenhoeck & Ruprecht, 149–177.

Schrape, Jan-Felix, 2015: Social Media, Massenmedien und Öffentlichkeit. Eine soziologische Einordnung. In: Imhof, Kurt et al. (Hg.): *Demokratisierung durch Social Media?* Wiesbaden: Springer VS, 199–212.

Schrape, Jan-Felix/Dickel, Sascha, 2015: Dezentralisierung, Demokratisierung, Emanzipation. Zur Architektur des digitalen Technikutopismus. In: *Leviathan* 43(3), 442–463.

Schrape, Jan-Felix, 2013: Neue Medien – alte Visionen. In: Donk, André/Becker, Rainer (Hg.): *Politik und Wissenschaft im Technikwandel*. Münster/Berlin: Lit, 85–100.

Schrape, Jan-Felix, 2011: Social Media, Massenmedien und gesellschaftliche Wirklichkeitskonstruktion. In: *Berliner Journal für Soziologie* 21(3), 407–429.

Schrape, Jan-Felix, 2011: Was ist die ‚Markenidentität' der Soziologie?. In: *Sozialwissenschaften und Berufspraxis* 34(2), 141–153.

Werle, Raymund/Troy, Irene, 2012: Wissen handelbar gemacht? Politik und Patente. In: *Politische Vierteljahresschrift* SH 46, 152–189.

Werle, Raymund, 2012: Institutions and Systems: Analysing Technical Innovation Processes from an Institutional Perspective. In: Johannes Bauer et al. (Eds.): *Innovation Policy and Governance in High-Tech Industries*. Heidelberg/Berlin: Springer, 23–47.