LINUP FRONT

2

Linux und freie Software

Inhalt

Lernziele

- Die Grundprinzipien von Linux und freier Software kennen
- Die gängigen FOSS-Lizenzen einordnen können
- Von den wichtigsten freien Anwendungsprogrammen gehört haben
- Von den wichtigsten Linux-Distributionen gehört haben

Vorkenntnisse

- Grundkenntnisse über Computer und Betriebssysteme (Kapitel 1)

2.1 Linux: Eine Erfolgsgeschichte

Im Sommer 1991 studierte Linus Torvalds, damals 21 Jahre alt, Informatik an der Technischen Universität von Helsinki (Finnland)[1]. Er hatte damals einen neuen 386-PC, den er ausprobieren wollte, und amüsierte sich damit, einen Terminal-Emulator zu schreiben, der ohne Betriebssystem auf der rohen Hardware lief und mit dem er auf den Unix-Rechner an der Universität zugreifen konnte. Aus diesem Programm wurde schließlich der erste Linux-Betriebssystemkern.

Unix hatte zu diesem Zeitpunkt schon gut 20 Jahre auf dem Buckel, aber war das Betriebssystem der Wahl an Universitäten und überall da, wo Forschung und Entwicklung betrieben wurde – die technisch-wissenschaftlichen »Workstations« der damaligen Zeit liefen praktisch alle mit verschiedenen Unix-Versionen.

Unix-Urzeit

Unix selbst hatte – fast wie Linux – als »Hobbyprojekt« von Ken Thompson und Dennis Ritchie bei den Bell Laboratories, dem Forschungsinstitut des US-Telekommunikationsgiganten AT&T, angefangen. Es mauserte sich schnell zu einem sehr nützlichen System, und da es zum größten Teil in einer höheren Programmiersprache (C) geschrieben war, konnte es relativ schnell von der ursprünglichen PDP-11 auf andere Rechnerplattformen portiert werden. Außerdem durfte AT&T in den 1970er Jahren keine Software verkaufen, so dass Unix ohne Support zu Selbstkosten »verschenkt« wurde – und da das System klein und übersichtlich war, wurde es zur beliebten Fallstudie in den Betriebssystem-Seminaren der meisten Universitäten.

Ende der 1970er portierten Studenten der University of California in Berkeley Unix auf die VAX, die Nachfolgeplattform der PDP-11, und bauten dabei verschiedene Verbesserungen ein, die als »BSD« (kurz für *Berkeley Software Distribution*) in Umlauf kamen. Diverse Ableger von BSD sind auch heute noch aktuell.

BSD

Minix

Zur Entwicklung der ersten Linux-Versionen benutzte Linus »Minix«, ein Unix-artiges Betriebssystem, das Andrew S. Tanenbaum an der Universität Amsterdam für Unterrichtszwecke geschrieben hatte. Minix war ziemlich simpel gehalten und außerdem nicht frei verfügbar, stellte also kein vollwertiges Betriebssystem dar – Abhilfe war offensichtlich nötig![2]

Am 25. August 1991 kündigte Linus sein Projekt öffentlich an und lud den Rest der Welt zur Mithilfe ein. Zu diesem Zeitpunkt funktionierte das System als alternativer Betriebssystemkern für Minix.

Einen richtigen Namen hatte das System damals noch nicht. Linus nannte es »Freax« (aus »Freak« und »Unix«); er hatte am Anfang kurz über »Linux« als Name nachgedacht, die Idee aber dann als zu selbstverliebt abgelehnt. Als Linus' System auf den FTP-Server der Universität hochgeladen wurde, benannte Linus' Kollege Ari Lemmke, dem der Name »Freax« nicht gefiel, es in eigener Regie in »Linux« um. Linus stimmte der Änderung später zu.

Linux stieß auf beträchtliches Interesse und viele Freiwillige entschlossen sich zur Mitarbeit. Im Dezember 1992 erschien Linux 0.99, die erste Version unter der GPL (Abschnitt 2.2.3) und durchaus ein ausgewachsenes Betriebssystem mit vollständiger (wenn auch simpler) Unix-Funktionalität.

Linux 2.0

Linux 2.0 kam Anfang 1996 heraus und führte einige wichtige Neuerungen ein, etwa die Unterstützung von Mehrprozessor-Rechnern und die Möglichkeit, Module dynamisch zur Laufzeit in den Linux-Kern zu laden – ein bedeutender Schritt hin zu benutzerfreundlichen Linux-Distributionen.

[1] Linus Torvalds ist ethnischer Finne (im September 2010 nahm er die US-amerikanische Staatsbürgerschaft an), aber Angehöriger der schwedischsprachigen Minderheit. Darum hat er einen aussprechbaren Namen.

[2] BSD stand damals noch nicht frei zur Verfügung; Linus sagte einmal, dass er, wenn es BSD schon gegeben hätte, nie mit Linux angefangen hätte.

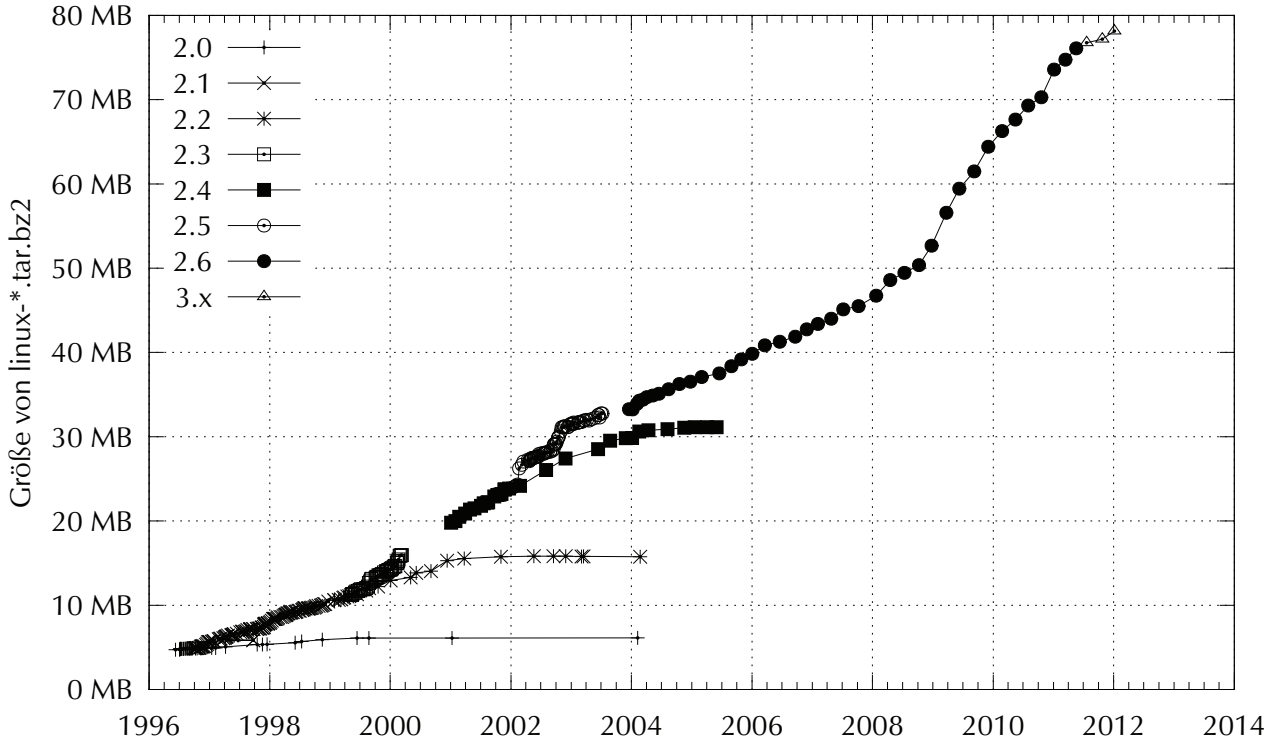

Bild 2.1: Die Weiterentwicklung von Linux, gemessen an der Größe von `linux-*.tar.bz2`. Jede Marke entspricht einer Linux-Version. In den 15 Jahren von Linux 2.0 bis Linux 3.2 hat der Umfang des komprimierten Linux-Quellcodes sich nahezu versechzehnfacht.

Ebenfalls neu in Linux 2.0 war »Tux«, der Pinguin, als offizielles Linux-Maskottchen. Linus Torvalds war in Australien von einem Pinguin angefallen worden, was ihn sehr beeindruckt hatte. Der sitzende Pinguin mit den gelben Füßen wurde von Larry Ewing gezeichnet und der Allgemeinheit zur Verfügung gestellt. Tux

Mit Linux 2.6 wurde Anfang 2004 der Entwicklungsprozess neu organisiert. Galten vorher Versionsnummern mit ungerader zweiter Stelle (etwa »2.3«) als Entwickler-Versionen und solche mit gerader zweiter Stelle (zum Beispiel »2.0«) als stabile Versionen für Endbenutzer, beschlossen die Linux-Entwickler, die Entwickler- und die stabilen Versionen künftig nicht mehr so stark divergieren zu lassen. Ab Linux 2.6 gab es keine separate Entwicklungslinie mehr, sondern Neuerungen fließen jeweils in die nächste Version ein und werden vor deren offizieller Freigabe möglichst ausgiebig getestet. Entwicklungsprozess

Man kann sich das ungefähr so vorstellen: Nach der Freigabe von Linux 2.6.37 sammelt Linus Neuerungen für den nächsten Linux-Kern, baut sie in seine offizielle Version ein und veröffentlicht diese dann als Linux 2.6.38-rc1 (für *release candidate* 1). Diese Version wird von diversen Leuten ausprobiert, und Reparaturen und Verbesserungen fließen in die Version 2.6.38-rc2 ein und so weiter. Irgendwann sieht der Code stabil genug aus, dass er als »Linux 2.6.38« offiziell freigegeben werden kann, und der Prozess wiederholt sich dann mit der Version 2.6.39.

Außer Linus' offizieller Version gibt es Linux-Versionen, die von anderen Entwicklern gepflegt werden. Zum Beispiel gibt es den »Staging Tree«, in dem neue Gerätetreiber »reifen« können, bis sie (nach einigen Runden von Verbesserungen) gut genug dafür sind, Linus für seine Version unterbreitet

zu werden. Einmal freigegebene Linux-Versionen bekommen meist noch eine Weile lang Reparaturen, so dass es auch Versionen wie 2.6.38.1, 2.6.38.2, … geben kann.

Linux 3.0

Im Juli 2011 erklärte Linus die damals gerade in Vorbereitung befindliche Version 2.6.40 summarisch zur Version 3.0 – nur um die Nummerierung zu vereinfachen, umwälzende Neuerungen gab es keine.

Release-Kandidaten heißen heute 3.2-rc1 und so weiter, die nach der Veröffentlichung reparierten Versionen entsprechend 3.1.1, 3.1.2, …

Das Projekt »Linux« ist auch heute nicht abgeschlossen. Linux wird ständig aktualisiert und erweitert, und zwar von Hunderten von Programmierern auf der ganzen Welt, denen inzwischen mehrere Millionen zufriedene private und kommerzielle Anwender gegenüberstehen. Man kann auch nicht sagen, dass das System »nur« von Studenten und anderen Amateuren entwickelt wird – viele Leute, die am Linux-Kern mitarbeiten, haben wichtige Posten in der Computerindustrie und gehören zu den fachlich angesehensten Systementwicklern überhaupt. Inzwischen läßt sich mit Berechtigung behaupten, dass Linux das Betriebssystem mit der breitesten Hardwareunterstützung überhaupt ist, nicht nur bezogen auf die Plattformen, auf denen es läuft (vom Smartphone bis zum Großrechner), sondern auch auf die Treiberunterstützung zum Beispiel auf der Intel-PC-Plattform. Linux dient auch als Test- und Forschungsplattform für neue Betriebssystem-Ideen in Industrie und Hochschule; es ist zweifellos eines der innovativsten derzeit verfügbaren Betriebssysteme.

Virtualisierung

Die Flexibilität von Linux macht es auch zum Betriebssystem der Wahl für Anwendungen wie Virtualisierung und »Cloud Computing«. Virtualisierung erlaubt es, auf einem tatsächlichen (»physikalischen«) Rechner mehrere bis viele »virtuelle« Rechner zu simulieren, die über ihr eigenes Betriebssystem verfügen und für dort laufende Programme so aussehen wie »echte« Rechner. Dies führt zu effizienterer Nutzung von Ressourcen und zu höherer Flexibilität: Die gängigen Infrastrukturen für Virtualisierung gestatten es, virtuelle Maschinen sehr schnell von einem physikalischen Rechner auf einen anderen zu »migrieren«, und als Betreiber einer entsprechenden Infrastruktur können Sie so sehr bequem auf Last-

Cloud Computing

situationen und Ausfälle reagieren. Cloud Computing ist darauf aufbauend die Idee, Rechenleistung nach Bedarf »auf Abruf« zur Verfügung zu stellen und Firmen damit die Möglichkeit zu geben, auf großangelegte Rechenzentren zu verzichten, die nur gelegentlich bei Anfragespitzen tatsächlich voll ausgelastet werden und ansonsten vor allem Kosten verursachen. Anbieter von Cloud-Computing gestatten ihren Kunden die Nutzung virtueller Maschinen über das Internet, bei Abrechnung auf Basis der tatsächlichen Nutzungsdauer, und das kann gegenüber dem Unterhalt eines »realen« Rechenzentrums zu erheblichen Einsparungen führen, insbesondere wenn man einbezieht, dass Sie als Kunde nicht nur die Anschaffungskosten, sondern auch die Personal-, Material- und Stromkosten für den 24/7-Betrieb des Rechenzentrums vermeiden.

Übungen

2.1 [2] Suchen Sie im Internet nach der berühmt-berüchtigten Diskussion zwischen Andrew S. Tanenbaum und Linus Torvalds, in der Tanenbaum sagt, Linus wäre mit etwas wie Linux bei ihm im Praktikum durchgefallen. Was halten Sie davon?

2.2 [1] Welche Versionsnummer hat der älteste Linux-Kern-Quellcode, den Sie noch finden können?

2.2 Frei oder Open Source?

2.2.1 Urheberrecht und »freie Software«

Im Mittelalter war die Vervielfältigung von Büchern und anderen Schriftstücken eine aufwendige Angelegenheit: Man musste jemanden finden, der des Schreibens mächtig war und die nötige Zeit hatte – was größere Projekte wie das Kopieren von Bibeln zur Domäne von Klöstern machte (die Mönche dort konnten schreiben und hatten jede Menge Zeit). Mit dem Aufkommen des Buchdrucks im 16. Jahrhundert entstand auf einmal ein Problem: Plötzlich wurde das Kopieren ein gutes Stück einfacher und billiger (jedenfalls sofern man eine Druckerpresse sein Eigen nannte), und eifrige Verleger nutzten das weidlich aus, um alles zu vervielfältigen, was sie für verkäuflich hielten. Die Autoren damals hatten eigentlich keine Rechte; sie konnten froh sein, wenn die Verleger ihnen irgendetwas dafür bezahlten, dass sie ihre Werke druckten. Das weitverbreitete Nachdrucken führte auch dazu, dass die Erstdrucker sich betrogen fühlten, wenn andere Drucker ihre Veröffentlichungen ohne Gegenleistung übernahmen. Deswegen wandten viele von ihnen sich an die Obrigkeit mit der Bitte darum, für bestimmte Werke ein Exklusivrecht zur Veröffentlichung zu erhalten. Dies kam der Obrigkeit durchaus zupass, da sie gerne den Überblick über die Druckwerke behalten wollte, die in ihrem Einflussbereich im Umlauf waren. Aus diesen »Privilegien« sowie weiteren Entwicklungen wie der (lobenswerten) Idee, auch den *Autoren* von Werken zum Beispiel das Recht auf ein Honorar einzuräumen, entstand das Konzept des Urheberrechts (bzw. im angelsächsischen Rechtsraum das verwandte Konzept des *copyright*).

Urheberrecht bedeutet nichts Anderes, als dass dem Urheber eines Werks, also dem Autor eines Buchs, Maler eines Bildes, … das Recht zuerkannt wird, als Einziger darüber zu verfügen, was mit dem Werk passiert. Er kann als Autor zum Beispiel einem Verlag das Recht abtreten (bzw., in der Praxis, gegen ein Honorar verkaufen), das Buch zu drucken und in Umlauf zu bringen; der Verlag kann damit Geld verdienen und der Autor muss sich nicht selbst um Vervielfältigung, Werbung, Vertrieb usw. kümmern, so dass beiden gedient ist.

Neben diesen »Verwertungsrechten« gibt es auch »moralische« Rechte, etwa das Recht, als Urheber eines Werks identifiziert zu werden. Diese moralischen Rechte kann ein Urheber nicht abtreten.

Im 20. Jahrhundert wurden die Konzepte des Urheberrechts weltweit (einigermaßen) vereinheitlicht und auch auf andere Werke wie Tonaufnahmen und Filme erweitert.

Mit der Ankunft von Computern und dem Internet gegen Ende des 20. Jahrhunderts wurde die Situation noch einmal gravierend verändert: Während das Urheberrecht früher vor allem dazu diente, Verleger vor anderen Verlegern zu schützen (Privatleute hatten selten die Mittel, Bücher, Schallplatten oder Filme in kommerziell merkbarem Umfang zu vervielfältigen), war es plötzlich jedem, der einen Computer besaß, möglich, digitale Inhalte (etwa Software oder Bücher, Musik oder Filme in digitaler Form) ohne Qualitätsverlust nahezu beliebig oft zu vervielfältigen und weiterzugeben – für Verleger, Musik-, Film- und Softwarefirmen eine Katastrophe, da die bestehenden Geschäftsmodelle, die auf dem Verkauf von physikalischen Artefakten wie Büchern oder CDs beruhten, ins Wanken kamen. Die »Inhalteindustrie« agitiert seitdem für eine Verschärfung der Urheberrechtsgesetze und höhere Strafen für »Raubkopierer« und versucht sich daran, Copyright-Sünder dingfest zu machen (mit wechselndem Erfolg).

Man spricht heute von »geistigem Eigentum« und meint damit nicht nur Urheberrechte, sondern auch Markenzeichen und Patente. Patente sollen die Erfinder von technischen Verfahren dafür belohnen, dass sie ihre Erfindungen dokumentieren und veröffentlichen, indem sie für eine Weile ein Exklusivrecht bekommen, diese Erfindungen auszunutzen (etwa indem sie Anderen gegen Geld gestatten, die Erfindungen anzuwenden). Markenzeichen

stellen sicher, dass niemand unerlaubt die Bekanntheit einer »Marke« ausnutzen darf, um seine eigenen Produkte unter die Leute zu bringen. Zum Beispiel sorgt ein Markenzeichen auf »Coca-Cola« dafür, dass nicht jeder, der eine dunkelbraune Zuckerbrühe brauen kann, diese als »Coca-Cola« in den Handel bringen darf. Die drei Formen von »geistigem Eigentum« sind verwandt, aber verschieden – bei Patenten geht es um Ideen, bei Urheberrechten um die konkrete Umsetzung von Ideen in Werke und bei Markenzeichen um die Verhinderung unsauberer Geschäftspraktiken.

Urheberrecht auf ein Werk erhält man automatisch, indem man das Werk verfasst oder erstellt – jedenfalls sofern das Werk eine gewisse minimale eigene Kreativität beinhaltet. Patente müssen beim Patentamt beantragt werden und werden auf Neuheit geprüft. Markenrechte kann man sich ebenfalls eintragen lassen oder erhält sie dadurch, dass man eine Marke eine Weile lang verwendet und als Anbieter eines Produkts von der Öffentlichkeit damit in Verbindung gebracht wird.

Auch Software für Computer (die man als eine Art von Schriftwerk ansehen kann und in der mitunter erhebliche Kreativität steckt) unterliegt dem Schutz des Urheberrechts. Das heißt, dass es grundsätzlich nicht erlaubt ist, ein Programm oder ein komplettes Softwarepaket einfach so zu kopieren, ohne dass der Urheber (der Programmierer oder dessen Firma) das ausdrücklich gestattet hat.

In der Frühzeit des Computers war es nicht üblich, Software zu verkaufen. Man bekam sie entweder kostenlos zu seinem Computer dazu (der war ja teuer genug – Millionen von Dollar oder Mark) oder man schrieb sie selbst. In den Universitäten der 1960er und der Heimcomputerszene der 1970er Jahre war es völlig normal, Programme zu tauschen und zu kopieren, und ein gewisser Bill Gates war 1976 völlig entsetzt darüber, dass sein BASIC-Interpreter für den MITS Altair 8800 zwar sehr verbreitet war und er jede Menge Lob dafür bekam, aber fast niemand es nötig fand, ihm den Kaufpreis dafür zu bezahlen! Das war von den Benutzern natürlich nicht korrekt, aber schon die *Idee*, dass Software etwas kosten sollte, war damals einfach so absurd, dass die wenigsten überhaupt auf den Gedanken kamen.

In den späten 1970er und 1980er Jahren wurde es mit der zunehmenden Verbreitung von Computern in Büros immer selbstverständlicher, Software nicht zu verschenken, sondern zu verkaufen. Nicht nur das – die Softwarefirmen verkauften nur den ausführbaren Maschinencode, nicht den Quellcode, an dem man hätte sehen können, wie die Software funktioniert, oder gar Veränderungen hätte machen können. Irgendwann war ein Punkt erreicht, wo Richard M. Stallman, damals Forscher am MIT, beschloss, dieser Entwicklung entgegenzuwirken und an einem System zu arbeiten, das die Kultur des Teilens, wie sie in den 1960er und 1970er Jahren üblich war, wieder in den Vordergrund stellen sollte. Dieses »GNU«-System[3] ist zwar immer noch nicht ganz fertig, aber viele Komponenten davon kommen heute in Linux-Systemen zum Einsatz.

freie Software

Auf Richard M. Stallman (oft kurz »RMS« genannt) geht das Konzept der »freien Software« zurück. »Frei« heißt in diesem Zusammenhang nicht »kostenlos«, sondern dass der Benutzer »frei« ist, verschiedene Dinge zu tun, die er mit proprietärer Software nicht könnte oder dürfte[4]. RMS nennt eine Software »frei«, wenn vier Bedingungen, die »vier Freiheiten«, erfüllt sind:

- Man muss das Programm für jeden beliebigen Zweck benutzen dürfen (Freiheit 0)

[3] »GNU« ist eine »rekursive« Abkürzung für *GNU's Not Unix*.

[4] Die *Free Software Foundation*, Stallmans Stiftung zur Förderung freier Software, nennt das *free as in speech, not as in beer*, also »frei wie freie Rede, nicht wie Freibier«.

- Man muss studieren können, wie das Programm funktioniert, und es an seine eigenen Bedürfnisse anpassen können (Freiheit 1)
- Man muss das Programm weitergeben dürfen, um seinem Nächsten zu helfen (Freiheit 2)
- Man muss das Programm verbessern und die Verbesserungen veröffentlichen dürfen, um der Allgemeinheit zu nutzen (Freiheit 3).

Zugang zum Quellcode ist Vorbedingung für die Freiheiten 1 und 3.

Die Idee der freien Software fand grundsätzlich Anklang, aber die Ziele von RMS und der *Free Software Foundation* (FSF) wurden doch oft missverstanden. Insbesondere Firmen störten sich an dem Wort »frei«, das sie trotz entsprechender Klarstellungen zu sehr mit dem Wort »kostenlos« verwechselten. Ende der 1990er Jahre gründeten Eric S. Raymond, Bruce Perens und Tim O'Reilly, die *Open Source Initiative* (OSI), deren Ziel es war, freier Software zu besserem und weniger ideologischem Marketing zu verhelfen. Die FSF war nicht begeistert von dieser »Verwässerung«, und die Kontroverse ist trotz der doch sehr ähnlichen Ziele von FSF und OSI auch heute noch nicht komplett beigelegt (wobei auch die ausgeprägten Egos einiger der maßgeblich beteiligten Personen eine Rolle spielen könnten).

Während das »frei« in »freie Software« die Verwechslung mit »kostenlos« nahelegt, kann man den »offengelegten Quellcode« in »Open Source Software« auch so interpretieren, dass der Quellcode zwar zu besichtigen ist, aber keine Änderung oder Weitergabe zuläßt – beides eigentlich Kernprinzipien auch der OSI. In diesem Sinne ist keiner der beiden Begriffe wirklich zu 100% treffend, so dass in der Szene oft von »FOSS« (kurz für *free and open-source software*) oder gar »FLOSS« (*free, libre and open-source software*, wobei das *libre* den Begriff der »Freiheit« unterstützen soll) die Rede ist.

FOSS
FLOSS

Wie kann man mit freier Software Geld verdienen, wenn jeder die Software ändern und weitergeben darf? Eine sehr berechtigte Frage. Hier sind ein paar Ideen für »Open-Source-Geschäftsmodelle«:

Geschäftsmodelle

- Sie können für die freie Software Zusatzleistungen wie Unterstützung, Dokumentation oder Training anbieten und sich dafür bezahlen lassen (für uns als Linup Front GmbH funktioniert das sehr gut, und auch das LPI kann anscheinend vom Linux-Zertifizierungsgeschäft einigermaßen leben).
- Sie können auf Anfrage kundenspezifische Weiterentwicklungen oder Erweiterungen erstellen und sich Ihre Arbeitszeit vergüten lassen (auch wenn das Ergebnis der Entwicklung dann Bestandteil der allgemein verfügbaren Version wird). Das funktioniert sogar für freie Software, die Sie nicht ursprünglich selber geschrieben haben.

  

  Im »traditionellen« Modell der proprietären Softwareentwicklung hat zunächst einmal der Hersteller ein Monopol auf Änderungen und Weiterentwicklungen. Als Kunde einer solchen Firma haben Sie ein Problem, wenn der Hersteller das Produkt abkündigt oder selbst verschwindet (etwa insolvent geht oder von seinem größten Konkurrenten aufgekauft wird), weil Sie dann mit Kosten und Mühe eine Software eingeführt haben, die keine Zukunft hat. Bei freier Software dagegen können Sie immer versuchen, jemanden zu finden, der statt der ursprünglichen Herstellerfirma die Unterstützung übernimmt – notfalls tun Sie sich dafür mit anderen Anwendern der Software zusammen, die genausowenig im Regen stehen wollen.

- Wenn Sie ein Softwarepaket anbieten, können Sie eine Version davon als FOSS vertreiben, die die Grundfunktionalität abdeckt, und hoffen, dass genug Leute, die von der FOSS-Version angelockt wurden, die proprietäre »Vollversion« kaufen, um wirklich etwas getan zu kriegen. (Der Jargon dafür ist *open core*.)

Dies ist eine zweischneidige Sache: Zum einen ist es natürlich schön, wenn es mehr freie Software gibt, aber zum anderen läuft es oft darauf hinaus, dass man die proprietäre Version *braucht*, weil wichtige Funktionen in der freien Version nicht zur Verfügung stehen und es zuviel Aufwand wäre, diese in eigener Regie »nachzurüsten«. Die »freie« Version hat dann in erster Linie die Rolle eines Werbeträgers, wenn die Herstellerfirma als modern und »open-source-freundlich« gelten möchte, ohne das aber wirklich zu meinen.

2.2.2 Lizenzen

Wie wird eine Software zur »freien« oder »Open-Source«-Software? Wir hatten gesagt, dass gewisse Rechte – etwa das Recht zur Vervielfältigung oder zur Veränderung eines Werks – dem Urheber des Werks vorbehalten sind, er diese Rechte aber auch an andere weitergegeben kann. Dies geschieht (wenn es geschieht) auf dem Weg einer »Lizenz«, also eines juristischen Textes, der festlegt, welche Rechte der Empfänger der Software durch das Kaufen, Herunterladen, … an der Software erwirbt.

Lizenz

Schon das Urheberrecht als solches erlaubt es dem Käufer (oder rechtmäßigem Herunterlader) einer Software, die Software zum Beispiel auf einem Computer zu installieren, aufzurufen und laufen zu lassen. Das resultiert einfach daraus, dass ihm die Software vom Urheber zugänglich gemacht wurde – schließlich hat es keinen Zweck, jemandem ein Programm zu verkaufen, das er anschließend nicht benutzen darf. (Umgekehrt hat der, der dem Programmautor Geld für das Programm gegeben hat, grundsätzlich sogar ein *Recht* darauf, als Gegenleistung das Programm benutzen zu dürfen.) Andere Aktionen, etwa das unkontrollierte Kopieren und Weitergeben oder das Verändern des Programms, schließt das Urheberrecht dagegen aus, und wenn der Programmautor jemandem das Recht dazu geben möchte, muss er es in die Lizenz schreiben.

EULA

Proprietäre Programme kommen oft mit einem »Endbenutzer-Lizenzabkommen« (*end-user license agreement* oder EULA), das der Käufer akzeptieren muss, bevor er die Software tatsächlich benutzen darf. Der Verkäufer der Software benutzt das EULA, um dem Käufer Dinge zu verbieten, die er gemäß dem Urheberrechtsgesetz eigentlich hätte – etwa die Software »gebraucht« weiterzuverkaufen oder in der Öffentlichkeit schlecht (oder überhaupt) über die Software zu reden. Ein solches EULA ist ein Vertrag, der von beiden Seiten akzeptiert werden muss, und die rechtlichen Hürden dafür liegen (zumindest in Deutschland) ziemlich hoch. Zum Beispiel müssen einem Software-Käufer die EULA-Bedingungen vor dem Kauf bekannt sein oder es muss zumindest möglich sein, die Software unbenutzt zurückzugeben, wenn dem Käufer die Bedingungen nicht zusagen.

Open-Source-Lizenzen

Die Lizenzen für freie und Open-Source-Software dagegen dienen dazu, dem Erwerber der Software Dinge zu erlauben, die er laut Urheberrechtsgesetz eigentlich sonst *nicht* dürfte. Sie versuchen in der Regel nicht, den *Gebrauch* der Software einzuschränken, sondern ordnen vor allem Fragen der Veränderung und Weitergabe der Software. In diesem Sinne stellen sie den Erwerber der Software nicht schlechter, als er bei irgendeinem Sachkauf zu erwarten hätte. Deshalb sind Freie-Software-Lizenzen im Gegensatz zu EULAs normalerweise keine Verträge, die der Empfänger der Software akzeptieren muss, damit sie wirksam werden, sondern einseitige Erklärungen des Softwareautors, und die von ihnen eingeräumten Rechte eine Art »Sonderbonus« zum einfachen Nutzungsrecht.

Inzwischen gibt es einen ganzen Zoo von Lizenzen, die die Anforderungen an freie oder Open-Source-Software erfüllen. Die bekannteste Freie-Software-Lizenz ist die *General Public License* (GPL) des GNU-Projekts von Richard M. Stallman, aber es gibt einige andere. Die OSI „zertifiziert“ Lizenzen, die ihrer Meinung nach

den Open-Source-Gedanken verkörpern, genau wie die FSF Lizenzen gutheißt, die die »vier Freiheiten« sichern. Welche das im Einzelfall sind, erfahren Sie auf den Web-Seiten dieser Organisationen.

Wenn Sie gedenken, ein freies oder Open-Source-Softwareprojekt zu starten, dann können Sie sich natürlich auch eine Lizenz überlegen, die den Anforderungen der FSF oder OSI genügt. Empfehlenswerter ist es jedoch, eine bestehende, bereits anerkannte Lizenz für das Projekt zu übernehmen. Zum einen müssen Sie dann nicht bei FSF oder OSI um die Anerkennung Ihrer Lizenz werben, und zum anderen sind die gängigen Lizenzen bereits juristisch hinreichend abgeklopft worden, um als einigermaßen wasserdicht gelten zu können – als Amateur auf dem Gebiet des Vertrags- oder Lizenzrechts könnten Sie durchaus wichtige Dinge übersehen, die Sie dann später in Schwierigkeiten bringen könnten.

FOSS und Urheberrecht

Es ist eine sehr wichtige Beobachtung, dass es den Proponenten freier oder Open-Source-Software in keiner Weise darum geht, das Urheberrecht für Software komplett abzuschaffen. Tatsächlich kann freie Software, so wie wir sie kennen, überhaupt nur funktionieren, weil es das Urheberrecht *gibt* und die Urheber von Software darum das Recht haben, Veränderung und Weitergabe der Software an Bedingungen zu koppeln wie dass der Empfänger der Software wiederum das Recht zur Veränderung und Weitergabe bekommt. Ohne Urheberrecht könnte jeder sich (wie zu den Zeiten Gutenbergs) beliebig bei der verfügbaren Software bedienen, und zentrale Grundsätze wie die »vier Freiheiten« wären in großer Gefahr, weil es möglich wäre, nur zu hamstern und nie zu teilen.

2.2.3 Die GPL

Der Linux-Kern und weite Teile dessen, was man sonst unter »Linux« versteht, stehen unter der *General Public License* (GPL). Die GPL wurde von RMS für das GNU-Projekt entworfen und soll sicherstellen, dass Software, die einmal unter der GPL steht, auch weiter unter der GPL bleibt (eine solche Lizenz nennt man auch *Copyleft*-Lizenz). Und das funktioniert ungefähr so:

- GPL-Software muss im Quellcode zur Verfügung stehen und darf für beliebige Zwecke benutzt werden.
- Es ist ausdrücklich erlaubt, diesen Quellcode zu ändern und in Originalform oder geänderter Form weiterzugeben, solange der Empfänger dieselben Rechte unter der GPL erhält.
- Es ist ebenfalls ausdrücklich erlaubt, GPL-Software in ausführbarer Form weiterzugeben oder sogar zu verkaufen. In diesem Fall muss aber der Quellcode (mitsamt den GPL-Rechten) mitgeliefert oder für einen bestimmten Zeitraum auf Anfrage zugänglich gemacht werden.

»Quellcode« heißt in diesem Zusammenhang »alles, was nötig ist, um die Software auf einem Computer zum Laufen zu bringen«. Was das im Einzelfall bedeutet, also ob zum Beispiel auch die kryptografischen Schlüssel dazugehören, die nötig sind, um auf einem entsprechend abgesicherten Rechner einen angepassten Linux-Kernel zu starten, ist Gegenstand hitziger Diskussionen.

Wenn jemand eine GPL-Software für Geld kauft, bekommt er damit natürlich das Recht, die Software nicht nur auf allen seinen Computern zu installieren, sondern sie auch zu kopieren und weiterzuverkaufen (unter der GPL). Das hat einerseits die Konsequenz, dass es nicht viel Sinn ergibt, GPL-Software „pro Rechner“ zu verkaufen, aber andererseits auch den angenehmen Nebeneffekt, dass es die Preise etwa für Linux-Distributionen halbwegs vernünftig hält.

- Wenn Sie ein neues Programm schreiben, das Teile eines GPL-Programms beinhaltet, müssen Sie das neue Programm als »abgeleitetes Werk« auch unter die GPL stellen.

  
  Auch hier erhitzen sich die Gemüter an einer Abgrenzung, wie viel und was für Teile von einem GPL-Programm übernommen sein müssen, damit ein anderes Programm als »abgeleitetes Werk« gilt. Wenn man der FSF glaubt, führt schon die Benutzung einer dynamisch ladbaren GPL-Bibliothek in einem Programm dazu, dass das Programm unter die GPL fällt, sogar wenn es selber überhaupt keinen Code der GPL-Bibliothek beinhaltet und darum im Sinne des Urheberrechts auch nicht »abgeleitet« sein kann. Wieviel hiervon auf Wunschdenken beruht und wieviel tatsächlich urheberrechtlich haltbar ist, bedarf grundsätzlich einer gerichtlichen Klärung.

Die GPL bezieht sich ausschließlich auf die Veränderung und Weitergabe der Software und nicht auf deren Einsatz.

GPLv3


Im Moment sind zwei Versionen der GPL in Gebrauch. Die neuere Version 3 (auch als »GPLv3« bezeichnet) wurde Ende Juni 2007 veröffentlicht und unterscheidet sich von der älteren Version 2 (auch »GPLv2«) durch Präzisierungen in Bereichen wie Softwarepatenten, der Kompatibilität mit anderen freien Lizenzen und der Einführung von Restriktionen dafür, Änderungen an der theoretisch »freien« Software von Geräten unmöglich zu machen, indem man sie durch spezielle Hardware ausschließt (die »Tivoisierung«, nach einem digitalen Videorekorder auf Linux-Basis, dessen Linux-Kern man nicht verändern oder austauschen kann). Die GPLv3 erlaubt ihren Benutzern auch das Hinzufügen weiterer Klauseln. – Die GPLv3 stieß nicht auf die uneingeschränkte Zustimmung der Szene, so dass viele Projekte, allen voran der Linux-Kernel, mit Absicht bei der einfacheren GPLv2 geblieben sind. Viele Projekte werden auch »unter der GPLv2 oder einer neueren Version der GPL« vertrieben, so dass Sie sich entscheiden können, welcher Version der GPL Sie bei der Weitergabe oder Änderung solcher Software folgen wollen.


Es gilt unter Entwicklern freier Software als guter Stil, Beiträge zu einem Projekt unter dieselbe Lizenz zu stellen, die das Projekt schon benutzt, und die meisten Projekte bestehen tatsächlich darauf, dass das zumindest für solchen Code gilt, der in die »offizielle« Version einfließen soll. Manche Projekte bestehen sogar auf *copyright assignments*, bei denen der Urheber des Codes seine Rechte an das Projekt (oder eine geeignete Organisation) abtritt. Dieser Schritt hat den Vorteil, dass urheberrechtlich nur das Projekt für den Code verantwortlich ist und Lizenzverstöße – die nur der Rechte-Inhaber verfolgen kann – leichter geahndet werden können. Ein entweder gewollter oder ausdrücklich unerwünschter Nebeneffekt ist auch, dass es einfacher möglich ist, die Lizenz für das ganze Projekt zu ändern, denn auch das darf nur der Rechte-Inhaber.


Im Falle des Linux-Betriebssystemkerns, wo ausdrücklich keine *copyright assignments* verlangt werden, ist eine Lizenzänderung praktisch sehr schwierig bis ausgeschlossen, da der Code ein Flickenteppich von Beiträgen von über tausend Autoren ist. Die Angelegenheit wurde im Zuge der Veröffentlichung der GPLv3 erörtert, und man war sich einig, dass es ein riesengroßes Projekt wäre, die urheberrechtliche Provenienz jeder einzelnen Zeile des Linux-Quellcodes zu klären und die Zustimmung der Autoren zu einer Lizenzänderung einzuholen. Manche Linux-Entwickler wären auch vehement dagegen, andere sind nicht mehr zu finden oder sogar verstorben, und der betreffende Code müsste durch etwas Ähnliches mit klarem Copyright ersetzt werden. Zumindest Linus Torvalds ist aber nach wie vor Anhänger der GPLv2, so dass das Problem sich in der Praxis (noch) nicht wirklich stellt.

GPL und Geld

Die GPL sagt nichts über den möglichen Preis des Produkts aus. Es ist absolut legal, dass Sie Kopien von GPL-Programmen verschenken oder auch Geld dafür verlangen, solange Sie auch den Quellcode mitliefern oder auf Anfrage verfügbar machen und der Empfänger der Software auch die GPL-Rechte bekommt. GPL-Software ist damit also nicht unbedingt »Freeware«.

Mehr Informationen erhalten Sie durch Lesen der GPL [GPL91], die übrigens jedem entsprechenden Produkt (auch Linux) beiliegen muss.

Die GPL gilt als konsequenteste der freien Lizenzen in dem Sinne, dass sie – wie erwähnt – sicherzustellen versucht, dass einmal unter der GPL veröffentlichter Code auch frei *bleibt*. Es haben schon öfters Firmen versucht, GPL-Code in ihre eigenen Produkte zu integrieren, die dann nicht unter der GPL freigegeben werden sollten. Allerdings haben diese Firmen bisher immer nach einer nachdrücklichen Ermahnung durch (meist) die FSF als Rechteinhaberin eingelenkt und die Lizenzbestimmungen eingehalten. Zumindest in Deutschland ist die GPL auch schon gerichtlich für gültig erklärt worden – einer der Linux-Kernelprogrammierer konnte vor dem Landgericht Frankfurt ein Urteil gegen die Firma D-Link (einen Hersteller von Netzwerkkomponenten, hier einem NAS-Gerät auf Linux-Basis) erwirken, in dem letztere zu Schadensersatz verklagt wurde, weil sie bei der Verbreitung ihres Geräts den GPL-Auflagen nicht genügte [GPL-Urteil06].

Warum funktioniert die GPL? Manche Firma, der die Anforderungen der GPL lästig waren, hat versucht, sie für ungültig zu erklären oder erklären zu lassen. So wurde sie in den USA schon als »unamerikanisch« oder »verfassungswidrig« apostrophiert; in Deutschland wurde versucht, das Kartellrecht heranzuziehen, da die GPL angeblich illegale Preisvorschriften macht. Die Idee scheint zu sein, dass Software, die unter der GPL steht, für jeden beliebig benutzbar ist, wenn mit der GPL irgendetwas nachweislich nicht stimmt. Alle diese Angriffe verkennen aber eine Tatsache: Ohne die GPL hätte niemand außer dem ursprünglichen Autor überhaupt das Recht, mit dem Code irgendetwas zu tun, da Aktionen wie das Weitergeben oder gar Verkaufen nach dem Urheberrecht nur ihm vorbehalten sind. Fällt die GPL weg, dann stehen alle anderen Interessenten an dem Code also viel schlechter da als vorher.

Ein Prozess, bei dem ein Softwareautor eine Firma verklagt, die seine GPL-Software vertreibt, ohne sich an die GPL zu halten, würde aller Wahrscheinlichkeit nach so aussehen:

Richter Was ist jetzt das Problem?

Softwareautor Herr Vorsitzender, die Beklagte hat meine Software vertrieben, ohne die Lizenz dafür einzuhalten.

Richter (zum Rechtsanwalt der Beklagten) Stimmt das?

An dieser Stelle kann die beklagte Firma »Ja« sagen und der Prozess ist im Wesentlichen vorbei (bis auf das Urteil). Sie kann auch »Nein« sagen, aber sie muss dann begründen, warum das Urheberrecht für sie nicht gilt. Ein unangenehmes Dilemma und der Grund, warum wenige Firmen sich diesen Stress machen und die meisten GPL-Streitigkeiten außergerichtlich geklärt werden.

Verstößt ein Hersteller proprietärer Software gegen die GPL (etwa indem er ein paar hundert Zeilen Quellcode aus einem GPL-Projekt in seine Software integriert), bedeutet das übrigens in keinem Fall, dass seine Software dadurch automatisch komplett unter die GPL fällt und offengelegt werden muss. Es bedeutet zunächst nur, dass er GPL-Code gegen dessen Lizenz vertrieben hat. Lösen kann der Hersteller das Problem auf verschiedene Arten:

- Er kann den GPL-Code entfernen und durch eigenen Code ersetzen. Die GPL wird dann irrelevant für sein Programm.

- Er kann mit dem Urheberrechtsinhaber des GPL-Codes verhandeln und zum Beispiel eine Zahlung von Lizenzgebühren vereinbaren (wenn dieser aufzutreiben ist und mitmacht). Siehe auch den Abschnitt über Mehrfachlizenzierung weiter unten.
- Er kann sein komplettes Programm *freiwillig* unter die GPL stellen und so den Anforderungen der GPL entsprechen (die unwahrscheinlichste Methode).

Unabhängig davon könnte für den vorher stattgefundenen Lizenzverstoß Schadensersatz verhängt werden. Der urheberrechtliche Status der proprietären Software bleibt davon allerdings unberührt.

2.2.4 Andere Lizenzen

Außer der GPL gibt es auch noch andere Lizenzen, die im FOSS-Umfeld üblich sind. Hier ist ein kleiner Überblick:

BSD-Lizenz Die BSD-Lizenz stammt aus dem Umfeld der Unix-Distribution der University of California in Berkeley und ist absichtlich sehr einfach gehalten: Der Empfänger der Software bekommt im Wesentlichen das Recht, mit der Software zu machen, was er will, solange er nicht den Eindruck erweckt, als stünde die Universität (oder im erweiterten Sinne der ursprüngliche Softwareautor) hinter ihm. Eine Haftung für das Programm wird so weit wie möglich ausgeschlossen. Der Lizenztext muss im Quellcode des Programms erhalten bleiben und – beim Vertrieb des Programms oder veränderter Versionen davon in ausführbarer Form – in der Dokumentation wiedergegeben werden.

Früher enthielt die BSD-Lizenz auch die Anforderung, dass in Werbematerial für Software oder Systeme, die BSD-Code enthielten, auf diesen Umstand aufmerksam gemacht werden musste. Diese Klausel wurde jedoch inzwischen entfernt.

Im Gegensatz zur GPL versucht die BSD-Lizenz nicht, den Quellcode der Software in der Öffentlichkeit zu halten. Wer sich BSD-lizenzierte Software besorgt, kann sie grundsätzlich in seine eigenen Programme integrieren und diese Programme nur in ausführbarer Form weitergeben (die GPL würde verlangen, dass auch der Quellcode weitergegeben werden muss).

Kommerzielle Softwarefirmen wie Microsoft oder Apple, die von GPL-Software gemeinhin nicht besonders begeistert sind, haben in der Regel kein Problem mit BSD-lizenzierter Software. Windows NT zum Beispiel verwendete eine ganze Weile lang den TCP/IP-Netzwerkcode von BSD (in angepasster Form), und auch große Teile des Betriebssystemkerns von OS X auf dem Macintosh beruhen auf BSD.

In der FOSS-Szene gehen seit langem die Meinungen darüber auseinander, ob nun die GPL oder die BSD-Lizenz »freier« ist. Einerseits kann man sagen, dass man mit BSD-lizenzierter Software als Empfänger mehr anfangen kann, dass also in absoluten Begriffen die BSD-Lizenz mehr Freiheit einräumt. Dem entgegen steht die Sicht der GPL-Verfechter, die sagen, dass es wichtiger ist, dass Code für alle frei bleibt, als dass er in proprietären Systemen verschwindet, und dass ein Kennzeichen größerer Freiheit der GPL darum darin besteht, dass diejenigen, die sich aus dem Vorrat der GPL-Software bedienen, auch verpflichtet werden, etwas zurückzugeben.

Apache-Lizenz Die Apache-Lizenz ähnelt der BSD-Lizenz darin, dass sie die Nutzung und Übernahme von lizenziertem Code gestattet, ohne (wie die GPL) zu verlangen, dass modifizierter Apache-lizenzierter Code wieder

der Öffentlichkeit zugänglich gemacht wird. Sie ist komplexer als die BSD-Lizenz, aber enthält auch Klauseln über die Nutzung von Patenten und Markenzeichen und andere Details.

Mozilla Public License (MPL) Die Mozilla-Lizenz (die zum Beispiel für den Firefox-Browser gilt) ist eine Mischung aus der BSD-Lizenz und der GPL. Sie ist eine »schwache Copyleft-Lizenz«, da sie einerseits verlangt, dass man Code, den man unter der MPL erhält, unter der MPL weitergeben muss (ähnlich wie bei der GPL), andererseits aber auch gestattet, dass man dem MPL-Code Code unter anderen Lizenzen hinzufügt, der dann nicht unter der MPL weitergegeben werden muss.

Creative Commons

Der Erfolg der FOSS-Gemeinde brachte den Juristen Lawrence (Larry) Lessig dazu, das Konzept auf andere Werke außer Software anzuwenden. Ziel war, den Fundus von Kulturgütern wie Büchern, Bildern, Musik, Filmen, … zu vergrößern, die anderen zur freien Übernahme, Veränderung und Weitergabe zur Verfügung stehen. Da die gängigen FOSS-Lizenzen sehr auf Software abgestimmt sind, wurden hierfür die *Creative-Commons*-Lizenzen entwickelt, die es Urhebern ermöglichen, ihre Werke kontrolliert der Allgemeinheit zu übergeben. Dabei können sie verschiedene Einschränkungen festlegen, wie dass das Werk nur unverändert weitergegeben werden darf, dass Änderungen erlaubt sind, aber (GPL-mäßig) Empfänger der Änderungen wieder Änderungen machen dürfen, oder dass eine kommerzielle Nutzung des Werks ausgeschlossen ist.

Public Domain

»Public Domain« beschreibt Kulturgüter, die *gemeinfrei* sind, also keinen Urheberrrechten mehr unterliegen. Während es in der angelsächsischen Rechtstradition möglich ist, ein Werk (also auch eine Software) ausdrücklich in die *public domain* zu stellen, geht das in Deutschland nicht, da das Konzept im deutschen Urheberrecht gar nicht existiert. Hierzulande werden Werke erst 70 Jahre nach dem Ableben des letzten Urhebers automatisch gemeinfrei. Bis das erste Computerprogramm auf diesem Weg allgemein nutzbar wird, wird es also noch ein bisschen dauern.

Es bestehen allerdings gute Chancen, dass überhaupt keine Werke, die nach ca. 1930 erstellt wurden, jemals gemeinfrei werden. In den USA jedenfalls wird die Copyright-Frist jedesmal vom Kongress verlängert, wenn die Gefahr besteht, dass eine gewisse Comic-Maus in die *public domain* übergeht. Der Rest der Welt zieht dann normalerweise bereitwillig nach. Es ist nicht ganz klar, warum sogar noch die *Urenkel* von Walt Disney dank der Kreativität des großen Zeichners Geld scheffeln müssen wie Dagobert Duck, aber wie so oft hängt das vor allem davon ab, wer die cleversten Lobbyisten hat.

Mehrere Lizenzen

Grundsätzlich kann der Inhaber des Urheberrechts für eine Software diese Software auch gleichzeitig unter mehreren Lizenzen vertreiben – etwa für FOSS-Entwickler unter der GPL und für Firmen, die ihren eigenen Quellcode nicht offenlegen möchten, unter einer proprietären Lizenz. Das lohnt sich natürlich vor allem für Bibliotheken, die andere Entwickler in ihre eigenen Programme integrieren. Wer proprietäre Software entwickeln möchte, kann sich dann von den Anforderungen der GPL »freikaufen«.

Übungen

2.3 [!1] Welche der folgenden Aussagen über die GPL stimmen und welche sind falsch?

1. GPL-Programme dürfen nicht verkauft werden.
2. GPL-Programme dürfen von Firmen nicht umgeschrieben und zur Grundlage eigener Produkte gemacht werden.
3. Der Urheber eines GPL-Programms darf das Programm auch unter einer anderen Lizenz vertreiben.

4. Die GPL gilt nicht, weil man die Lizenz erst zu sehen bekommt, nachdem man das Programm schon hat. Damit Lizenzbestimmungen gültig werden, muss man sie vor dem Erwerb der Software sehen und ihnen zustimmen können.

2.4 [2] Vergleichen Sie die »vier Freiheiten« der FSF mit den *Debian Free Software Guidelines* (`http://www.debian.org/social_contract#guidelines`) des Debian-Projekts (siehe Abschnitt 2.4.4). Welche Definition freier Software gefällt Ihnen besser und warum?

2.3 Wichtige freie Programme

2.3.1 Überblick

Linux ist ein leistungsfähiges und elegantes Betriebssystem, aber das schönste Betriebssystem ist nichts wert ohne Programme, die darauf laufen. In diesem Abschnitt präsentieren wir eine Auswahl der wichtigsten freien bzw. Open-Source-Programme, die auf typischen Linux-PCs zu finden sind.

Dass irgendein Programm nicht enthalten ist, heißt nicht, dass wir das Programm nicht gut finden. Der Platz ist halt begrenzt, und wir versuchen, vor allem diejenige Software abzudecken, die das LPI in den Prüfungsrichtlinien erwähnt hat (*just in case*).

2.3.2 Büro- und Produktivitätsprogramme

Die meisten Computer werden wahrscheinlich für »Büroanwendungen« wie das Schreiben von Briefen und Memos, Diplom- und Doktorarbeiten, die Auswertung von Daten mit Tabellenkalkulations- und Grafikprogrammen und Ähnliches verwendet. Außerdem verbringen viele Anwender große Teile ihrer Computer-Zeit im Internet oder mit dem Lesen und Schreiben von E-Mail. Kein Wunder, dass es auf diesem Gebiet jede Menge gute freie Software gibt.

Die meisten der Programme in diesem Abschnitt stehen nicht nur für Linux zur Verfügung, sondern auch für Windows, OS X oder gar andere Unix-Varianten. Das macht es möglich, Benutzer langsam auf ein FOSS-Umfeld »umzugewöhnen«, indem Sie zum Beispiel auf einem Windows-Rechner zuerst LibreOffice, Firefox und Thunderbird anstelle von Microsoft Office, Internet Explorer und Outlook installieren, bevor Sie das Betriebssystem selbst durch Linux ersetzen. Wenn Sie das geschickt anfangen[5], merken die Benutzer den Unterschied vielleicht gar nicht.

OpenOffice.org ist seit Jahren das Flaggschiff-Produkt der FOSS-Gemeinde, wenn es um große Büropakete geht. Es hat vor vielen Jahren als »StarOffice« angefangen und wurde schließlich von Sun aufgekauft und (in einer leicht abgespeckten Version) als freie Software in Umlauf gebracht. OpenOffice.org enthält alles, was man so in einem Office-Paket erwartet – Textverarbeitung, Tabellenkalkulation, Präsentationsprogramm, Geschäftsgrafik, Datenbank, … –, und kann auch einigermaßen mit den Datenformaten des großen Mitbewerbers von Microsoft umgehen.

LibreOffice Nachdem Sun von Oracle übernommen wurde und die Zukunft von OpenOffice.org zunächst in den Sternen stand, taten sich einige der wesentlichen OpenOffice.org-Entwickler zusammen und veröffentlichten ihre eigene Version von OpenOffice.org unter dem Namen »LibreOffice«. Die beiden Pakete werden im Moment nebeneinander her entwickelt (Oracle hat

[5] Zum Beispiel könnten Sie Ihren Schäflein das aktuelle Ubuntu als »Windows-8-Beta« unterschieben …

OpenOffice.org der Apache Software Foundation übergeben) – sicherlich kein optimaler Zustand, aber es ist unklar, ob und wie es zu einer »Wiedervereinigung« kommen wird.


Die meisten großen Linux-Distributionen liefern inzwischen LibreOffice aus, das aktiver weiterentwickelt und – vor allem – aufgeräumt wird.

Firefox ist inzwischen der beliebteste Web-Browser und wird von der Mozilla Foundation in Umlauf gebracht. Firefox ist sicherer und effizienter als der bisherige »Platzhirsch« (Microsofts Internet Explorer), kann mehr und entspricht eher den Standards für das World Wide Web. Außerdem gibt es einen großen Zoo an Erweiterungen, mit dem Sie Firefox an Ihre eigenen Anforderungen anpassen können.

Chromium ist die FOSS-Variante des Google-Browsers »Chrome«. Chrome hat in der letzten Zeit angefangen, Firefox mächtig Konkurrenz zu machen – auch Chrom{e,ium} ist ein leistungsfähiger, sicherer Browser mit zahlreichen Erweiterungen. Vor allem die Google-Webangebote sind auf den Google-Browser abgestimmt und laufen dort besonders gut.

Thunderbird ist ein Mail-Programm von der Mozilla Foundation. Es hat große Teile seines Unterbaus mit dem Firefox-Browser gemeinsam und bietet – wie Firefox – einen reichen Fundus an Erweiterungen für verschiedene Zwecke.

2.3.3 Grafik- und Multimedia-Werkzeuge

Grafik und Multimedia ist seit jeher die Domäne der Macs (auch wenn die Windows-Seite da auch nette Programme zu bieten hat). Zugegebenermaßen fehlt unter Linux noch ein echtes Äquivalent zu Programmen wie Adobes Photoshop, aber die Software, die es gibt, ist auch nicht zu verachten.

The GIMP ist ein Programm, mit dem Sie Fotos und ähnliche Vorlagen bearbeiten können. Es kann noch nicht ganz mit Photoshop mithalten (zum Beispiel fehlen Funktionen bei der Druckvorstufe), aber ist für viele Anwendungen absolut brauchbar und bietet zum Beispiel bei der Erstellung von Grafiken für das World Wide Web ein paar Eigenschaften, die Photoshop nicht genauso bequem zur Verfügung stellt.

Inkscape Wenn The GIMP das Photoshop von Linux ist, dann ist Inkscape das Pendant zu Illustrator – eine leistungsfähige Software zur Erstellung vektorbasierter Grafiken.

ImageMagick ist ein Softwarepaket, mit dem Sie fast beliebige Grafikformate in fast beliebige andere umwandeln können. Außerdem erlaubt es Ihnen die skriptgesteuerte Manipulation von Grafiken auf nahezu endlose verschiedene Arten. Genial für Webserver und andere Umgebungen, wo mit Grafiken gearbeitet werden muss, ohne dass Maus und Monitor zum Einsatz kommen können.

Audacity dient als Mehrspurtonband, Mischpult und Schneidestation für Audiodaten aller Art und ist auch auf Windows und dem Mac populär.

Cinelerra und andere Programme wie KDenlive oder OpenShot sind »nichtlineare Videoeditoren«, die Videos von digitalen Camcordern, TV-Karten oder Webcams schneiden, vertonen, mit Effekten versehen und in verschiedenen Formaten (von YouTube bis zur DVD) ausgeben können.

Blender ist nicht nur ein leistungsfähiger Videoeditor, sondern erlaubt auch das fotorealistische »Rendern« von dreidimensionalen bewegten Szenen und ist damit das angesagte Werkzeug zum Erstellen von Animationsfilmen in professioneller Qualität.

An dieser Stelle ein Hinweis darauf, dass heutzutage kein Hollywood-Blockbuster mehr entsteht, ohne dass Linux mit im Spiel ist: Die Spezialeffekt-Renderfarmen der großen Studios laufen inzwischen alle unter Linux.

2.3.4 Server-Dienste

Ohne Linux wäre das Internet nicht wiederzuerkennen: Die Hunderttausende Server von Google laufen genauso mit Linux wie die Handelssysteme der meisten großen Börsen der Welt (die Deutsche Börse genau wie die Börsen von London und New York), weil nur mit Linux die dafür nötige Leistung realisierbar ist. Tatsächlich wird die meiste Internet-Software heute zuerst auf Linux entwickelt, und auch an Universitäten findet die Forschung in diesem Bereich natürlich auf der quelloffenen Linux-Plattform statt.

Apache ist mit Abstand der beliebteste Web-Server auf dem Internet – mehr als die Hälfte aller Webseiten laufen auf einem Apache-Server.

Es gibt natürlich auch andere gute Web-Server auf Linux – etwa Nginx oder Lighttpd –, aber Apache bleibt der populärste.

MySQL und PostgreSQL sind frei verfügbare relationale Datenbankserver. MySQL eignet sich vor allem für Webseiten, während PostgreSQL ein innovativer und hochleistungsfähiger Datenbankserver für alle möglichen Anwendungen ist.

Postfix ist ein sicherer und extrem leistungsfähiger Mailserver, der für alle Anforderungen vom »Home-Office« bis zum großen Provider oder DAX-Unternehmen gerüstet ist.

2.3.5 Infrastruktur-Software

Auch innerhalb eines lokalen Netzes macht ein Linux-Server eine gute Figur: Zuverlässig, schnell und wartungsarm kann man ihn installieren und vergessen (bis auf die regelmäßigen Sicherungskopien natürlich!).

Samba macht einen Linux-Rechner zu einem Server für Windows-Clients, der Plattenplatz und Drucker für alle Windows-Rechner im Netz zugänglich macht (Linux-Rechner übrigens auch). Mit dem neuen Samba 4 kann ein Linux-Rechner sogar als Active-Directory-Domänencontroller dienen und macht darum einen Windows-Server überflüssig. Stabilität, Effizienz und eingesparte Lizenzkosten sind sehr überzeugende Argumente.

NFS ist das Unix-basierte Äquivalent zu Samba und erlaubt anderen Linux- und Unix-Rechnern im Netz den Zugriff auf die Platten eines Linux-Servers. Linux unterstützt das moderne NFSv4 mit erhöhter Leistung und Sicherheit.

OpenLDAP dient als Verzeichnisdienst zur Strukturierung mittlerer und großer Netze und erlaubt durch seine leistungsfähigen Features zur Verteilung und Replikation von Daten eine hohe Redundanz und Geschwindigkeit bei der Abfrage und Aktualisierung.

DNS und DHCP gehören zur grundlegenden Netzwerkinfrastruktur. Mit BIND unterstützt Linux den Referenz-DNS-Server, und der ISC-DHCP-Server kann mit BIND kooperieren, um auch in sehr großen Netze Clients mit Netzparametern wie IP-Adressen zu versorgen. Dnsmasq ist ein bequem zu administrierender DNS- und DHCP-Server für kleine Netze.

2.3.6 Programmiersprachen und Entwicklung

Linux war von Anfang an ein System von Entwicklern für Entwickler. Entsprechend stehen Übersetzer und Interpreter für alle wichtigen Programmiersprachen zur Verfügung – die GNU-Compilerfamilie deckt zum Beispiel C, C++, Objective C, Java, Fortran und Ada ab. Natürlich werden die populären Skriptsprachen wie Perl, Python, Tcl/Tk, Ruby, Lua oder PHP unterstützt, und auch »Exoten« wie Lisp, Scheme, Haskell, Prolog und Ocaml sind Bestandteil vieler Linux-Distributionen.

Eine reichhaltige Auswahl von Editoren und Hilfswerkzeugen macht die Softwareentwicklung zum Vergnügen. Der Standard-Editor `vi` steht ebenso zur Verfügung wie professionelle Umgebungen zur Programmierung, etwa GNU Emacs oder Eclipse.

Linux taugt außerdem als Entwicklungssystem für »Embedded-Systeme«, also Rechner, die als Bestandteil von Konsumgütergeräten oder spezialisierte »Appliances« entweder selbst unter Linux laufen oder eigene Betriebssysteme mitbringen. Es ist leicht, unter Linux auf dem PC eine Compilerumgebung zu installieren, die zum Beispiel Maschinencode für ARM-Prozessoren erzeugt. Daneben ist Linux auch gut geeignet zur Softwareentwicklung für Android-Smartphones, die von Google mit professionellen Werkzeugen unterstützt wird.

Übungen

2.5 [!1] Von welchen FOSS-Programmen haben Sie schon gehört? Welche davon haben Sie schon selber benutzt? Finden Sie sie besser oder schlechter als proprietäre Alternativen? Wenn ja, warum? Wenn nein, warum nicht?

2.4 Wichtige Linux-Distributionen

2.4.1 Überblick

Wenn jemand etwas sagt wie »Ich habe Linux auf meinem Rechner«, meint er damit in der Regel nicht (nur) Linux, den Betriebssystemkern, sondern eine komplette Softwareumgebung auf der Basis von Linux. Dazu gehören meist die Shell `bash` und die Kommandozeilenwerkzeuge aus dem GNU-Projekt, der X.org-Grafikserver und eine grafische Arbeitsumgebung wie KDE oder GNOME, Produktivitätsprogramme wie LibreOffice, Firefox oder The GIMP und viele andere nette Software aus dem vorigen Abschnitt. Es ist natürlich grundsätzlich möglich, sich alle diese Komponenten selbst aus dem Internet zusammenzusuchen, aber die meisten Linux-Anwender benutzen eine vorgefertigte Software-Sammlung oder »Linux-Distribution«.

Die ersten Linux-Distributionen kamen Anfang 1992 heraus – allerdings wird keine Distribution aus der damaligen Zeit noch aktiv weiterentwickelt, und sie sind zum größten Teil vergessen. Die älteste Distribution, an der tatsächlich noch gearbeitet wird, ist Slackware, die zuerst im Juli 1993 erschien.

Es gibt eine Vielzahl von Linux-Distributionen mit verschiedenen Zielen und Ansätzen und unterschiedlicher Organisationsstruktur. Einige Distributionen werden von Firmen veröffentlicht und mitunter sogar nur gegen Geld verkauft, während andere Distributionen von Freiwilligen-Teams oder Einzelpersonen zusammengestellt werden. Im Folgenden diskutieren wir kurz die wichtigsten Distributionen für den allgemeinen Gebrauch.

Wenn wir Ihre Lieblings-Distribution hier nicht erwähnen, hat das weniger damit zu tun, dass wir sie nicht ausstehen können, sondern mehr damit, dass unser Platz und unsere Zeit begrenzt sind und wir uns (leider!) auf das Wesentliche konzentrieren müssen. Wenn eine Distribution hier nicht

vorkommt, bedeutet das also nicht, dass sie schlecht oder nutzlos ist, sondern nur, dass sie hier nicht vorkommt.

Die Webseite »DistroWatch« (`http://distrowatch.com/`) listet die wichtigsten Linux-Distributionen auf und dient als Anlaufstelle für Neuigkeiten rund um Distributionen. Im Moment sind dort 317 Distributionen (in Worten: dreihundertsiebzehn!) gelistet, wobei diese Zahl, wenn Sie diese Zeilen lesen, wahrscheinlich schon nicht mehr stimmt.

2.4.2 Red Hat

Red Hat (`http://www.redhat.com/`) wurde 1993 unter dem Namen »ACC Corporation« als Vertriebsfirma für Linux- und Unix-Zubehör gegründet. 1995 kaufte der Firmengründer, Bob Young, das Geschäft von Marc Ewing, der 1994 eine Linux-Distribution unter dem Namen »Red Hat Linux« veröffentlicht hatte, und firmierte zu »Red Hat Software« um. 1999 ging Red Hat an die Börse und ist seitdem wahrscheinlich das größte Unternehmen, das ausschließlich auf Linux und Open-Source-Software setzt. Es ist in den »Standard & Poor's 500« vertreten, einem Aktienindex, der (ähnlich dem DAX in Deutschland) als Indikator für die US-amerikanische Wirtschaft dient.

Aus dem ursprünglichen Privatkundengeschäft hat Red Hat sich zurückgezogen (das letzte »Red Hat Linux« erschien im April 2004) und vertreibt heute unter dem Namen »Red Hat Enterprise Linux« (RHEL) eine Distribution für den professionellen Einsatz in Firmen. RHEL wird pro Server lizenziert, wobei Sie damit nicht für die Software bezahlen – die steht wie üblich unter der GPL oder unter anderen FOSS-Lizenzen –, sondern für den Zugriff auf zeitnahe Aktualisierungen und Unterstützung bei Problemen. RHEL ist vor allem für den Einsatz in Rechenzentren gedacht und erlaubt (mit den entsprechenden Zusatzpaketen) zum Beispiel den Aufbau von fehlertoleranten »Clustern«.

»Fedora« (`http://www.fedoraproject.org/`) ist eine von Red Hat maßgeblich gesteuerte Distribution, die als »Testumgebung« für RHEL dient. Neue Software und Ideen werden zuerst in Fedora umgesetzt, und was sich bewährt, taucht früher oder später in RHEL auf. Im Gegensatz zu RHEL wird Fedora nicht verkauft, sondern nur zum kostenlosen Download angeboten; das Projekt wird von einem Komitee geleitet, dessen Mitglieder teils von der Entwicklergemeinde gewählt und teils von Red Hat bestimmt werden. (Der Vorsitzende des Komitees wird von Red Hat bestimmt und hat ein Vetorecht.) Der Fokus auf aktuelle Software und neue Ideen macht für viele Fedora-Anwender den Reiz der Distribution aus, auch wenn das häufige Aktualisierungen bedeutet. Für Anfänger und den Einsatz auf Servern, die zuverlässig laufen sollen, ist Fedora weniger gut geeignet.

Da Red Hat seine Software konsequent unter FOSS-Lizenzen wie der GPL vertreibt, ist es grundsätzlich möglich, ein System zu betreiben, das dem aktuellen RHEL entspricht, ohne dafür Lizenzgebühren an Red Hat zahlen zu müssen.

CentOS
Scientific Linux

Es gibt Distributionen wie CentOS (`http://www.centos.org/`) oder Scientific Linux (`https://www.scientificlinux.org/`), die im Wesentlichen auf RHEL basieren, aber alle Markenzeichen von Red Hat entfernen. Das heißt, dass man im Wesentlichen identische Software bekommt, allerdings ohne die Unterstützung von Red Hat.

Insbesondere CentOS ist so nahe an RHEL dran, dass Red Hat Ihnen bei Bedarf Unterstützung für Ihre CentOS-Rechner verkauft, ohne dass Sie erst RHEL dort installieren müssen.

2.4.3 SUSE

Gründung

Die Firma SUSE wurde 1992 unter dem Namen »Gesellschaft für Software- und

System-Entwicklung« als Unix-Beratungshaus gegründet und schrieb sich entsprechend zuerst »S.u.S.E.«. Eines ihrer Produkte war eine an den deutschen Markt angepasste Version der Linux-Distribution Slackware (von Patrick Volkerding), die ihrerseits von der ersten kompletten Linux-Distribution, *Softlanding Linux System* oder SLS, abgeleitet war. S.u.S.E. Linux 1.0 erschien 1994 und differenzierte sich langsam von Slackware, indem beispielsweise Eigenschaften von Red Hat Linux wie die RPM-Paketverwaltung oder die `/etc/sysconfig`-Datei übernommen wurden. Die erste S.u.S.E.-Linux-Version, die nicht mehr wie Slackware aussah, war die 4.2 von 1996. SuSE (die Punkte wurden irgendwann weggelassen) eroberte bald die Marktführerschaft im deutschsprachigen Raum und veröffentlichte SuSE Linux als »Kiste« in zwei Geschmacksrichtungen, »Personal« und »Professional«; letztere war merklich teurer und enthielt unter anderem mehr Software aus dem Server-Bereich.

Übernahme durch Novell

Im November 2003 kündigte die US-amerikanische Softwarefirma Novell an, die SuSE für 210 Millionen Dollar übernehmen zu wollen; der Handel wurde dann im Januar 2004 perfekt gemacht. (Bei dieser Gelegenheit wurde auch das »U« groß gemacht.) Im April 2011 wurde Novell mitsamt SUSE von der Firma Attachmate übernommen. Attachmate ist in Bereichen wie Terminalemulation, Systemmonitoring und Anwendungsintegration aktiv und hat sich im Linux- und Open-Source-Bereich bisher nicht besonders hervorgetan. Seitdem wird Novell in zwei Einheiten weitergeführt, eine davon ist die SUSE.

Attachmate

Wie Red Hat bietet SUSE ein »Unternehmens-Linux« an, den *SUSE Linux Enterprise Server* (SLES, `http://www.suse.com/products/server/`), der RHEL darin ähnelt, dass er relativ selten erscheint und einen langen Lebenszyklus von 7–10 Jahren verspricht. Daneben gibt es mit dem *SUSE Linux Enterprise Desktop* (SLED) auch eine Distribution, die für den Einsatz auf Arbeitsplatzrechnern gedacht ist. SLES und SLED unterscheiden sich in der Paketauswahl; der Fokus bei SLES liegt mehr auf Serverdiensten und bei SLED eher auf interaktiver Software.

Auch SUSE hat inzwischen wie Red Hat den Schritt gemacht, die »Privatkunden«-Distribution zu öffnen und als »openSUSE« (`http://www.opensuse.org/`) frei verfügbar zu machen (früher gab es die Distribution immer erst mit einigen Monaten Verzögerung zum Download). Im Gegensatz zu Red Hat bietet SUSE aber nach wie vor eine »Kiste« an, die zusätzlich proprietäre Software enthält. Im Gegensatz zu Fedora ist openSUSE eine ernstgemeinte Plattform, die aber trotzdem einen relativ kurzen Lebenszyklus hat.

YaST

Bezeichnendes Merkmal der SUSE-Distributionen ist der »YaST«, ein umfassendes grafisch orientiertes Systemverwaltungswerkzeug.

2.4.4 Debian

Im Gegensatz zu den beiden großen Linux-Distributionsfirmen Red Hat und Novell/SUSE ist das **Debian-Projekt** (`http://www.debian.org/`) ein Zusammenschluss von Freiwilligen, die es sich zum Ziel gesetzt haben, eine hochwertige Linux-Distribution unter dem Namen *Debian GNU/Linux* frei zur Verfügung zu stellen. Das Debian-Projekt wurde am 16. August 1993 von Ian Murdock angekündigt; der Name ist eine Zusammensetzung seines Vornamens mit dem seiner damaligen Freundin (jetzt Ex-Frau) Debra (und wird darum »Debb-Ian« ausgesprochen). Inzwischen umfasst das Projekt über 1000 Freiwillige.

Debian-Projekt

Grundlage

Grundlage von Debian sind drei Dokumente:

- Die *Debian Free Software Guidelines* (DFSG) definieren, welche Software im Sinne des Projekts als „frei“ gilt. Das ist wichtig, denn nur DFSG-freie Software kann Teil der eigentlichen Debian-GNU/Linux-Distribution sein. Das Projekt vertreibt auch nichtfreie Software, diese ist jedoch auf den Distributionsservern strikt von der DFSG-freien Software getrennt: Letztere steht in

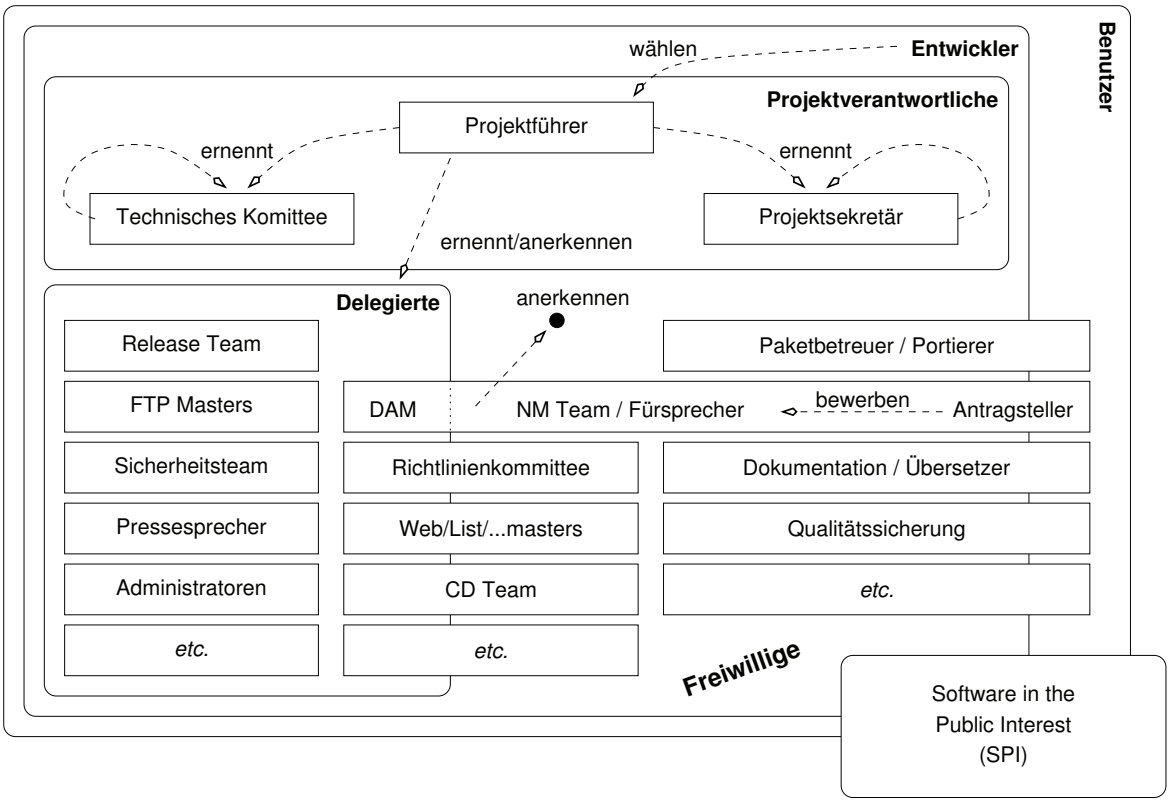

Bild 2.2: Organisationsstruktur des Debian-Projekts. (Grafik von Martin F. Krafft.)

einem Unterverzeichnis `main`, erstere in `non-free`. (Es gibt auch noch ein Mittelding namens `contrib`; dort findet sich Software, die für sich genommen DFSG-frei wäre, aber nicht ohne andere nichtfreie Komponenten funktioniert.)

- Der *Social Contract* (»Gesellschaftsvertrag«) beschreibt die Ziele des Projekts.
- Die *Debian Constitution* (»Verfassung«) beschreibt die Organisation des Projekts (siehe Bild 2.2).

Versionen

Zu jedem Zeitpunkt existieren mindestens drei Versionen von Debian GNU/Linux: In den `unstable`-Zweig werden neue oder korrigierte Versionen von Paketen eingebracht. Tauchen in einem Paket keine gravierenden Fehler auf, wandert es nach einer gewissen Wartezeit in den `testing`-Zweig. In gewissen Abständen wird der Inhalt von `testing` »eingefroren«, gründlich getestet und schließlich als `stable` freigegeben. Ein häufig geäußerter Kritikpunkt an Debian GNU/Linux sind die langen Zeiträume zwischen `stable`-Versionen; dies wird allerdings von vielen auch als Vorteil empfunden. Debian GNU/Linux wird vom Projekt ausschließlich zum Download zur Verfügung gestellt; Datenträger sind von Drittanbietern erhältlich.

Ableger-Projekte

Debian GNU/Linux ist durch seine Organisation, die weitgehende Abwesenheit kommerzieller Interessen und die saubere Trennung von freier und nichtfreier Software eine gute Grundlage für Ableger-Projekte. Einige populäre solche Projekte sind Knoppix (eine „Live-CD“, die es möglich macht, Linux auf einem PC zu testen, ohne es zuerst installieren zu müssen), SkoleLinux (heute »Debian/EDU«, ein speziell auf die Anforderungen von Schulen ausgerichtetes Linux) oder kommerzielle Distributionen wie Xandros. Auch Limux, das Münchner Desktop-Linux, basiert auf Debian GNU/Linux.

2.4.5 Ubuntu

Der wahrscheinlich populärste Debian-Ableger ist Ubuntu (`http://www.ubuntu.com/`), das von der britischen Firma Canonical Ltd. des südafrikanischen Unternehmers Mark Shuttleworth angeboten wird. (»Ubuntu« ist ein Wort aus der Zulu-Sprache und steht in etwa für »Menschlichkeit gegenüber anderen«.) Das Ziel von Ubuntu ist es, auf der Basis von Debian GNU/Linux ein aktuelles, leistungsfähiges und verständliches Linux anzubieten, das in regelmäßigen Abständen erscheint. Dies wird zum Beispiel dadurch erreicht, dass Ubuntu im Gegensatz zu Debian GNU/Linux nur drei statt über zehn Rechnerarchitekturen unterstützt und sich auf eine Teilmenge der in Debian GNU/Linux angebotenen Software beschränkt.

Ziel von Ubuntu

Ubuntu basiert auf dem `unstable`-Zweig von Debian GNU/Linux und verwendet in weiten Teilen dieselben Werkzeuge etwa zur Softwareverteilung, allerdings sind Debian- und Ubuntu-Softwarepakete nicht notwendigerweise miteinander kompatibel. Ubuntu erscheint in einem ziemlich regelmäßigen 6-Monats-Rhythmus, wobei alle zwei Jahre eine »LTS«-, also *long-term-support*-Version, herauskommt, für die Canonical Updates über einen Fünf-Jahres-Zeitraum verspricht.

Einige Ubuntu-Entwickler sind auch im Debian-Projekt aktiv, so dass es einen gewissen Austausch gibt. Andererseits sind nicht alle Debian-Entwickler begeistert von den Abkürzungen, die Ubuntu zuweilen im Namen des Pragmatismus nimmt, wo Debian vielleicht nach tragfähigeren, aber aufwendigeren Lösungen suchen würde. Ubuntu fühlt sich auch nicht im selben Maße der Idee der freien Software verpflichtet; während alle Infrastrukturwerkzeuge von Debian (etwa das Verwaltungssystem für Fehlerberichte) als freie Software zur Verfügung stehen, ist das für die von Ubuntu nicht immer der Fall.

Ubuntu vs. Debian

Ubuntu will nicht nur ein attraktives Desktop-System anbieten, sondern auch im Server-Bereich mit den etablierten Systemen wie RHEL oder SLES konkurrieren, also stabile Distributionen mit langem Lebenszyklus und guter Wartung anbieten. Es ist nicht klar, wie Canonical Ltd. auf lange Sicht Geld zu verdienen gedenkt; einstweilen wird das Projekt vor allem aus Mark Shuttleworths Schatulle unterstützt, die seit dem Verkauf seiner Internet-Zertifizierungsstelle Thawte an Verisign gut gefüllt ist …

Ubuntu vs. SUSE/Red Hat

2.4.6 Andere

Außer den genannten Distributionen gibt es noch viele weitere, etwa Mandriva Linux (`http://www.mandriva.com/en/linux/`) oder Turbolinux (`http://www.turbolinux.com/`) als kleinere Konkurrenten von Red Hat und SUSE, Gentoo Linux (`http://www.gentoo.org/`) als Distribution mit Fokus auf den Quellcode, zahlreiche »Live-Systeme« für verschiedene Zwecke von der Firewall bis hin zur Spiele- oder Multimedia-Plattform sowie sehr kompakte Systeme, die als Router, Firewall oder Rettungssystem einsetzbar sind.

Erwähnenswert vielleicht einfach aufgrund der Anzahl der »installierten Systeme« ist Android, das man mit etwas Wohlwollen durchaus als »Linux-Distribution« ansehen kann. Android besteht aus einem Linux-Betriebssystemkern mit einer Benutzerumgebung von Google auf der Basis von Googles Version von Java (»Dalvik«) anstatt der üblichen auf GNU, X, KDE usw. basierenden Benutzerumgebung, die die Grundlage der meisten »normalen« Distributionen bildet. Ein Android-Smartphone oder -Tablet präsentiert sich dem Benutzer also völlig anders als ein typischer Linux-PC unter openSUSE oder Debian GNU/Linux, ist aber dennoch im Grunde seines Wesens ein Linux-System.

Android

Während die meisten Android-Anwender ihr System vorinstalliert auf dem Telefon oder Tablet kaufen und dann nicht mehr ändern, ist es bei den meisten Android-basierten Geräten durchaus möglich (notfalls mit Tricks), eine alternative Android-»Distribution« zu installieren, von denen es inzwi-

schen auch etliche gibt. Für viele Geräte ist das die einzige Möglichkeit, aktualisierte Android-Versionen zu bekommen, wenn der Geräte-Hersteller oder der Telefondienst-Anbieter es nicht nötig finden, eine offizielle neue Version zu veröffentlichen.

2.4.7 Unterschiede und Gemeinsamkeiten

Gemeinsamkeiten

Obwohl es eine Unzahl an Distributionen gibt, verhalten zumindest die »großen« Distributionen sich in der täglichen Arbeit recht ähnlich. Das liegt zum einen daran, dass sie die gleichen grundlegenden Programme benutzen – beispielsweise ist der Kommandozeileninterpreter fast immer die `bash`. Zum anderen gibt es Standards, die dem Wildwuchs entgegenzuwirken versuchen. Zu nennen sind hier der *Filesystem Hierarchy Standard* (FHS) oder die *Linux Standard Base* (LSB), die versucht, eine einheitliche »Basisversion« von Linux zu definieren und die Versionsstände von Werkzeugen und Bibliotheken festzuschreiben, damit Drittanbieter es leichter finden, ihre Software für eine Vielzahl von Linux-Distributionen zu vertreiben.

Leider war die LSB nicht der durchschlagende Erfolg, mit dem man ursprünglich gerechnet hatte – sie wurde oft als Ansatz missverstanden, die Innovation in Linux zu verlangsamen oder zu stoppen und die Diversität zu verrringern (wobei es in den meisten Distributionen möglich ist, eine LSB-Umgebung parallel zu und unabhängig von der eigentlichen Umgebung für distributionseigene Software zur Verfügung zu stellen), während die Software-Drittanbieter, die eigentlich angesprochen werden sollten, es unter dem Strich vorzogen, ihre Pakete für die wichtigen »Enterprise-Distributionen« wie RHEL und SLES zu »zertifizieren« und nur diesen Plattformen Unterstützung zu gewähren – es ist also absolut nicht ausgeschlossen, dass SAP oder Oracle zum Beispiel auch auf Debian GNU/Linux laufen (oder zum Laufen zu bringen sind), aber die Kosten für große kommerzielle Softwarepakete wie diese sind so, dass die Lizenzgebühren für RHEL oder SLES keinen merkbaren Unterschied im Gesamtbild machen.

Paketformate

Ein merkbarer Punkt, wo die Distributionen sich unterscheiden, ist die Methode, mit der Softwarepakete verwaltet, also eingespielt und entfernt werden, und daraus resultierend das Dateiformat der fertigen Pakete in der Distribution. Hier gibt es zur Zeit zwei konkurrierende Ansätze, und zwar den von Debian GNU/Linux (»`deb`«) und den ursprünglich von Red Hat entwickelten (»`rpm`«). Wie üblich ist keiner der beiden dem anderen eindeutig überlegen, aber jeder von ihnen hat genug Alleinstellungsmerkmale, damit seine Anhänger nicht ohne weiteres bereit sind, umzusatteln. Der `deb`-Ansatz wird von Debian GNU/Linux, Ubuntu und anderen Debian-Ablegern verwendet, während Red Hat, SUSE und diverse andere von diesen abgeleitete Distributionen auf `rpm` setzen.

Abhängigkeiten

Beiden Ansätzen gemein ist eine umfassende Verwaltung von »Abhängigkeiten« zwischen Softwarepaketen, die dabei hilft, die Konsistenz des Systems zu sichern und zum Beispiel das Entfernen von Softwarepaketen verhindert, auf deren Anwesenheit sich andere Pakete im System verlassen (oder umgekehrt deren Installation erzwingt, wenn ein anderes Paket, das gerade installiert wird, sie benötigt und sie nicht schon vorhanden sind).

Während Windows- und OS-X-Anwender es gewöhnt sind, sich Software aus den verschiedensten Quellen zusammenzusuchen[6], versuchen zumindest die »großen« Distributionen wie Debian, openSUSE oder Ubuntu ihren Benutzern eine möglichst umfassende Softwareauswahl direkt über die Paketverwaltungswerkzeuge zur Verfügung zu stellen. Diese erlauben den Zugriff auf »Repositories«, also Paketlagerstätten, der Distributionen und

Repositories

[6] Wobei Apple und auch Microsoft derzeit vehement dabei sind, das von Smartphones bekannte Konzept des zentralen »App Stores« auch in ihren PC-Betriebssystemen zu etablieren.

gestatten zum Beispiel eine (mehr oder weniger komfortable) Suche nach bestimmten Softwarepaketen, die dann bequem – gegebenenfalls mit ihren Abhängigkeiten – über das Netz installiert werden können.

Es ist wichtig, anzumerken, dass es nicht notwendigerweise möglich ist, Pakete zwischen Distributionen auszutauschen, selbst wenn diese dasselbe Paketformat (`deb` oder `rpm`) verwenden – in den Paketen stecken diverse Annahmen darüber, wie eine Distribution aufgebaut ist, die weit über das Paketformat hinausgehen. Völlig ausgeschlossen ist es nicht (Debian GNU/Linux und Ubuntu sind einander zum Beispiel ähnlich genug, dass es unter Umständen durchaus klappen kann, ein für Debian gedachtes Paket auf einem Ubuntu-System zu installieren oder umgekehrt), aber als Faustregel gilt, dass die Gefahr eines Misserfolgs umso größer ist, je tiefer ein Paket im System verankert ist – ein Paket, das nur ein paar ausführbare Programme und ihre Dokumentation mitbringt, wirft weniger Probleme auf als ein Systemdienst, der in die Startsequenz integriert werden muss. Lassen Sie im Zweifelsfall die Finger von Experimenten mit ungewissem Ausgang.

Tabelle 2.1 zeigt eine Übersichtstabelle der wichtigsten Linux-Distributionen. Für mehr Informationen sollten Sie DistroWatch oder die Webseiten der einzelnen Distributionen heranziehen.

Tabelle 2.1: Vergleich der wichtigsten Linux-Distributionen (Stand: Februar 2012)

	RHEL	Fedora	SLES	openSUSE	Debian	Ubuntu
Hersteller	Red Hat	Red Hat + Comm.	SUSE	SUSE + Comm.	Debian-Projekt	Canonical + Comm.
Zielgruppe	Untern.	Geeks	Untern.	Privat	Unt/Priv	Unt/Priv
Kosten?	ja	nein	Support	nein	nein	nein
Erstveröfftl.	2003	2003	2000	2006	1993	2004
Rhythmus	3–4 J.	ca. 6 Mon.	3–4 J.	8 Mon.	ca. 2 J.	6 Mon.
Lebensdauer	10 J.	ca. 1 J.	7 J.	18 Mon.	3–4 J.	5 J. (LTS)
Plattformen	6	2	5	2	10	3
Pakete (ca.)	3000	26.000	?	14.650	29.050	37.000
Paketformat	`rpm`	`rpm`	`rpm`	`rpm`	`deb`	`deb`
Live-Medium?	?	ja	nein	ja	ja	ja

Zusammenfassung

- Die erste Version von Linux wurde von Linus Torvalds entwickelt und als »freie Software« im Internet zur Verfügung gestellt. Heute wirken Hunderte von Entwicklern weltweit an der Aktualisierung und Erweiterung des Systems mit.
- Freie Software erlaubt Anwendern den Einsatz zu beliebigen Zwecken, die Einsichtnahme in und die Veränderung des Codes sowie die Weitergabe originalgetreuer oder modifizierter Versionen.
- Lizenzen für freie Software geben dem Empfänger der Software Rechte, die er sonst nicht hätte, während Lizenzverträge für proprietäre Software versuchen, den Empfänger zum Verzicht auf Rechte zu bewegen, die er sonst hätte.
- Die GPL ist eine der populärsten Lizenzen für freie Software.
- Andere gängige Lizenzen für freie Software sind die BSD-Lizenz, die Apache- oder die Mozilla Public License. Creative-Commons-Lizenzen sind gedacht für andere Kulturgüter außer Software.
- Es gibt eine große Auswahl freier und Open-Source-Programme für die unterschiedlichsten Anwendungen.
- Es gibt sehr viele verschiedene Linux-Distributionen. Zu den bekanntesten gehören Red Hat Enterprise Linux und Fedora, SUSE Linux Enterprise Server und openSUSE, Debian GNU/Linux und Ubuntu.

Literaturverzeichnis

GPL-Urteil06 Landgericht Frankfurt am Main. »Urteil 2-6 0 224/06«, Juli 2006. http://www.jbb.de/urteil_lg_frankfurt_gpl.pdf

GPL91 Free Software Foundation, Inc. »GNU General Public License, Version 2«, Juni 1991. http://www.gnu.org/licenses/gpl.html

TD01 Linus Torvalds, David Diamond (Hg.) *Just for Fun: Wie ein Freak die Computerwelt revolutionierte*. Hanser Fachbuch, 2001. ISBN 3-446-21684-7.