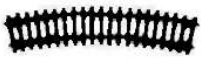
Ulrich Hansen

Zwischen Modelleisenbahn und Todesstern

Die Hacker-Ethik und die Entwicklung von Freier Software und Open Source



Ulrich Hansen, geb. 1967, Journalist. Studium an der Universität Dortmund, im Hauptfach Journalistik, im Nebenfach Technik-, Innovations- und Organisationssoziologie.

Studienarbeit an der Universität Dortmund

Institut für Journalistik
Fachbereich 15: Sprach- u. Literaturwissenschaften, Journalistik und Geschichte
Betreuung: Prof. Dr. Claus Eurich

Mainz, Februar 2005

(c) Ulrich Hansen, Mainz 2005.

Die zitierten Textausschnitte, Fotos und Abbildungen können eigenen urheberrechtlichen Bestimmungen unterliegen.

**„Denn, um es endlich auf einmal herauszusagen,
der Mensch spielt nur, wo er in voller Bedeutung des Wortes Mensch ist,
und er ist nur da ganz Mensch, wo er spielt.“**

Friedrich Schiller (1795)

**„Ein Hacker ist ein Mensch, der mit seiner Kaffeemaschine
auch eine Suppe kochen kann.“**

Wau Holland, Chaos Computer Club

Inhaltsverzeichnis

Einleitung 1
Begriffe und Schreibweisen 3
Motivation 4
1 Die Hacker-Ethik 5
1.1 Der Begriff "Hack" 5
1.2 Die ersten Hacker 6
1.3 Die Regeln der Hacker-Ethik 10
1.3.1 Freier Zugang zu Computern 11
1.3.2 Informationsfreiheit 12
1.3.3 Mißtrauen gegen Autorität 14
1.3.4 Meritokratie 17
1.3.5 Programmieren als Kunst 18
1.3.6 Technischer Optimismus 20
1.4 Zusammenfassung 24
2 Die Free-Software-Bewegung 25
2.1 Die Kommerzialisierung von Software 25
2.2 Stallman und der Bruch mit der Hacker-Ethik am MIT 32
2.2.1 Erste Legende: Der Kampf gegen Paßwörter 34
2.2.2 Zweite Legende: Der Ärger über einen Druckertreiber 36
2.2.3 Dritte Legende: Der "Symbolics-Krieg" 36
2.3 Das GNU-Projekt 39
2.3.1 Die Bedeutung von Unix 42
2.3.2 Der Verlauf des GNU-Projekts 45
2.3.3 Die GNU General Public License (GPL) 45
2.3.3.1 Die vier Freiheiten der GPL 47
2.3.3.2 "Copyleft" - die Pflichten der GPL 48
2.3.3.3 Die Bedeutung der GPL 50
2.3.4 Die Free Software Foundation 52
2.4 Zusammenfassung 54
3 GNU/Linux und Open Source 55
3.1 Das Versagen der "Unix-Hacker" 56
3.2 Linux 61
3.2.1 Linus Torvalds und die Entstehung von Linux 62
3.2.2 Die Freigabe von Linux 64
3.2.3 Die Entstehung der Linux-Community 67
3.2.4 Die Linux-Entwickler 71
3.3 Die Organisation der Projekte 77
3.3.1 Hierarchie 77
3.3.2 Management 79
3.3.3 Machtbegrenzung 82
3.4 Die Koordination der Projekte 84
3.4.1 Meritokratie als Entlohnungsform 86
3.4.2 Motivation der Mitarbeiter 90
3.4.2.1 Eigennutz 90

3.4.2.2 Lernen ... 91
3.4.2.3 Altruismus ... 92
3.4.2.4 Spaß ... 93
3.4.3 Identitätsstiftung durch Organisationskultur ... 94
3.4.3.1 Religion ... 95
3.4.3.2 Star Wars ... 99
3.4.3.3 Hacking ... 101
3.5 Der Begriff "Open Source" ... 102
3.5.1 Die Kommerzialisierung von GNU/Linux ... 103
3.5.2 Die Open-Source-Bewegung ... 106
3.6 Zusammenfassung ... 112
Schlußbemerkung ... 115
Der Einsatz freier Software in Medien ... 115
Die Freigabe von journalistischen Produkten ... 117
Open-Source-Journalismus ... 119
Bibliographie ... 123
Anhang ... I
Das Schwein guckt ins Uhrwerk oder: Was ist Quellcode? ... I
Interview mit Joseph Weizenbaum ... III
Nachweis der Diagramme ... V
GNU GENERAL PUBLIC LICENSE ... VI

Einleitung

Die Anhänger von Freier Software bzw. von Open-Source nehmen für sich eine hohe gesellschaftliche Bedeutung in Anspruch. Die Entscheidung der Münchner Stadtverwaltung, in ihren Büros von dem Betriebssystem Microsoft Windows auf die freie Software GNU/Linux umzusteigen, vergleichen sie mit dem Fall der Berliner Mauer.[1] Die gesetzliche Förderung von offenem oder geschlossenem Code ist für sie eine Frage von Rechtsstaat oder „Geheimregierung“[2] und für Richard Stallman, den Begründer der Bewegung, geht es schlicht um die Verteidigung der Freiheit.[3] Auch wenn der Tonfall gelegentlich etwas schrill[4] gerät, so wird der Anspruch doch auch von anderen gesellschaftlichen Kräften wie UNESCO[5], Attac[6] und der Partei Bündnis 90/Die Grünen[7] unterstützt.

Diese hohe gesellschaftliche Bedeutung scheint allerdings der deutschen Öffentlichkeit nicht sehr bekannt zu sein, wie eine Geschichte aus dem bayerischen Wahlkampf 2003 illustrieren mag: Damals warb die SPD-Politikerin und Programmiererin Monica Lochner-Fischer mit dem Plakat „Mehr Linux, mehr Freiheit“ für ihren Wiedereinzug in den Landtag – eine Anspielung auf die genannte GNU/Linux-Entscheidung im SPD-regierten München. Die Folge: Presseberichte, rund hundert E-Mails täglich von GNU/Linux-Fans sowie eine Einladung der Firma Microsoft.[8] Ihr CSU-Gegenkandidat im Wahlkreis München-Schwabing, Ludwig Spaenle, zeigte deutlich weniger elektronisches Engagement. „Das Internetangebot ist noch im Aufbau“[9] hieß es auf seiner Homepage, im August 2004

---

1 So geschehen durch den Vorstandsvorsitzenden der Firma Suse Linux AG, Richard Seibt, am 29.05.03. Vgl. NEFF, Berthold: David Linux besiegt Goliath Microsoft. Die Windows-Verkäufer verpokern sich mit Preisnachlässen. Neuer Anbieter hofft auf Signalwirkung. Süddeutsche Zeitung, 30.05.03, S. 40 (München).

2 LESSIG, Lawrence: Code und andere Gesetze des Cyberspace. Berlin: Berlin-Verlag, 2001, S. 391.

3 Vgl. STALLMAN, Richard Matthew: Free Software, Free Society. Selected Essays. Boston: GNU-Press, 2002, S. 30.

4 Ein Beispiel für diesen Tonfall ist die Presseerklärung der Free Software Foundation Europe (FSFE) zur Entscheidung des EU-Ministerrats über Softwarepatente: „Europa ist im Begriff, sich endgültig vom Ziel der Staats- und Regierungschefs zu verabschieden, bis 2010 ‘wettbewerbsfähigste wissensbasierte Region’ zu werden und hat zudem zum wiederholten Male demokratisch versagt. Es ist schade, daß der Zweckoptimismus und das Vertrauen in die deutsche Regierung etwas voreilig gewesen zu sein scheinen. Gestern war kein guter Tag für Europa und Deutschland!“ Vgl. JAKOBS, Joachim: Softwarepatente: Ein schlechter Tag für Europa und für Deutschland. Pressemitteilung der FSFE, 19.05.2004. http://mailman.fsfeurope.org/pipermail/press-release-de/2004q2/000025.html [Aufgerufen am 06.08.2004].

5 Im Jahr 1998 durch den damaligen UNESCO-Direktor für Information und Informatik, Philippe QUÉAU. Vgl. QUÉAU, Philippe: In search of the common good. The information society and archives. In: CITRA 1998: Access to information. Technological challenges. Proceedings of the thirty-third International Conference of the Round Table on Archives, Paris 1999, S. 17-26. Zitiert nach der Online-Veröffentlichung durch die UNESCO: http://www.unesco.org/webworld/points_of_views/queau_2.html [Aufgerufen am 08.08.2004].

6 Bei Attac Deutschland wird das Konzept von Free Software innerhalb der Arbeitsgemeinschaft „Wissensallmende und Freier Informationsfluß“ propagiert. Vgl.: http://www.attac.de/wissensallmende/ [Aufgerufen am 12.08.2004].

7 Auf einer Internetseite der Bundespartei heißt es: „BÜNDNIS 90/DIE GRÜNEN haben sich auf dem wirtschaftspolitischen Kongreß [am 28. und 29.05.04; Anm. v. UH] in Erfurt für den verstärkten Einsatz von Open-Source-Produkten und die Ablehnung von Softwarepatenten ausgesprochen. Open-Source-Software steht für die kooperative Weiterentwicklung von Software gegen wirtschaftliche Monokultur.“ http://www.gruene-partei.de/cms/themen_arbeit_wirtschaft/dok/27/27668.fuer_offene_quellen.htm [Aufgerufen am 12.08.04].

8 BIELICKI, Jan: Rote Äpfel und ein Fahrrad. Monica Lochner-Fischer (SPD) wirbt mit Obst und Linux – Ludwig Spaenle (CSU) lockt mit lokalen Themen. Süddeutsche Zeitung, 10.09.03, S. 47 (München).

9 http://www.ludwig-spaenle.de/ [Aufgerufen am 06.08.2004]

genauso wie im September 2003[10] und noch ein Jahr nach der Wahl dokumentierte er dort sein Interesse an dem schnellebigen Medium mit den Worten: „Liebe Mitbürgerinnen, liebe Mitbürger, die Wahl zum Bayerischen Landtag steht in Kürze bevor.“[11] Geschadet hat es ihm nicht: Spaenle gewann sogar noch hinzu und siegte mit 42,2 Prozent über Lochner-Fischer (29,8).[12] Das Thema trieb die Schwabinger nicht an die Urnen – die Lederhose schien ihnen näher zu stehen als der Laptop.

Wahlentscheidungen drücken gesellschaftliche Präferenzen aus, journalistische Berichterstattung auch: Sie ist ein Indikator für den Stellwert eines Themas. Es ist die Aufgabe der Medien, Themen von allgemeiner Bedeutung zu erkennen und darüber zu berichten.[13] Freie Software und Open Source gehören offensichtlich nicht dazu. Gerade mal knapp 400 Artikel sind von 1994 bis 2003 in FAZ und SZ zum Thema erschienen[14] und eine (oberflächliche) Untersuchung zeigt, daß GNU/Linux dort – ohne die reinen Techniktests zu berücksichtigen – vor allem im Kontext eines Themas auftaucht: Als Konkurrent von Microsoft. Die soziale oder rechtliche Bedeutung von GNU/Linux und Freier Software stand dagegen nur bei 55 Artikeln im Vordergrund.

In der Fernsehberichterstattung sieht es noch düsterer aus, nimmt man einen Beitrag im ZDF heute-journal vom 06.07.2004 zum Maßstab: Darin werden die Nutzer freier Software als naiv und fortschrittsfeindlich charakterisiert und in die Nähe von Raubkopierern gerückt, wogegen Siemens und der Bundeskanzler als Verteidiger geistigen Eigentums dargestellt werden.[15] Auch hier keine Worte über GNU/Linux, das durch die im Beitrag diskutierte Einführung von Softwarepatenten in seiner Existenz bedroht ist. Die Ironie daran: GNU/Linux wird auch im ZDF eingesetzt – und zwar als Serversystem und sendungsrelevant für eben dieses heute-journal, in dem der Bericht lief.[16]

Es fällt schwer, den Redakteuren einen Vorwurf zu machen. Freie Software besitzt für sie im Alltag genausowenig Relevanz wie für Ludwig Spaenle: Sie läuft nicht auf ihrem Organizer, nicht auf ihrem Laptop und auch nicht auf ihrem Bürorechner (sofern sie nicht bei der *taz* arbeiten). Und daß freie Software bei jedem Zugriff auf das Internet, bei jeder

---

10 Vgl. BIELICKI, a.a.O.

11 Vgl. Website des Politikers: http://www.ludwig-spaenle.de/ [Aufgerufen am 06.08.2004].

12 Sie zog dann, wie in der Wahlperiode zuvor, über einen Listenplatz in den Landtag ein.

13 So heißt es etwa im ersten Satz der „Entschließung zur Informationsvielfalt der Zeitungen mit Alleinstellung“, verabschiedet vom Deutschen Presserat am 5./6.06.1973: „Es gehört zu einer verantwortungsvollen publizistischen Arbeit, aus allen Lebensgebieten von öffentlichem Interesse unvoreingenommen und unter gewissenhafter Abwägung der Bedeutung der Nachrichten oder der Berichte zu informieren; das gilt auch für Auffassungen, die die Zeitung selbst nicht teilt.“ Zitiert nach: MEYN, Hermann: Massenmedien in der Bundesrepublik Deutschland. Berlin: Edition Colloquium, 1994, S. 57.

14 Anhand der Einträge in der ZDF-Zeitungsdatenbank. Dort sind seit 1996 jährlich jeweils 30.000 bis 50.000 Artikel beider Zeitungen im Volltext erfaßt. Auswahlkriterium für die Aufnahme ins Archiv ist die Eigenleistung der Zeitungen – übernommene Agenturmeldungen sind also nicht gespeichert. Gesucht wurde im Volltext der Artikel nach den Stichworten „*Linux* or (Open Source*) or (Open-Source*) or Stallman or (Free Software)“. Die Stichworte sollten damit alle verzeichneten Erwähnungen von Linux oder Open Source oder dem GNU-Projekt von Richard Stallman finden.

15 Der Text des Beitrags ist dokumentiert unter: http://kwiki.ffii.org/HeuteText040706De [Aufgerufen am 25.01.2005].

16 Siehe Schlußbemerkung.

E-Mail aktiv wird, wissen die meisten nicht.[17] Und ehrlich gefragt: Wen interessiert schon, was in den virtuellen Wirren des Internets mit einer E-Mail passiert, so lange sie beim Adressaten nur ankommt? Die Post als Thema war für Journalisten noch nie besonders sexy.

Die technische Seite von Freier Software steht auch bei dieser Arbeit nicht im Mittelpunkt. Statt dessen soll hier versucht werden, eine aktuelle, umfassende, aber verständliche Einführung in die soziale und ethische Entwicklung dieser Bewegung zu geben, ohne in technischen Jargon zu verfallen bzw. in den naiven oder gar missionarischen Tonfall, der im Internet und auch in mancher Fachliteratur gang und gäbe ist.

In einem Ausblick soll dann der Frage nachgegangen werden, welche Bedeutung die Philosophie der „Freiheit von Information“ für Journalisten haben könnte – und welche Rolle die neuen Copyright- und Veröffentlichungsformen dabei spielen könnten.

Begriffe und Schreibweisen

Der Begriff „Open Source“ hat seit Ende der 90er Jahre den Begriff „Freie Software“ zur Beschreibung freier und quelloffener Programme weitgehend abgelöst. Über die Unterschiede zwischen beiden gibt es aber in der Szene heftige Diskussionen: Aus politischen und historischen Gründen gilt der Begriff „Freie Software“ vielen als der korrektere; der Begriff „Open Source“ ist dagegen sehr viel bildstärker und kann im Englischen nicht als „Gratis-Software“ mißverstanden werden.

Die Versuche, aus einer vermeintlichen Objektivität heraus beide Begriffe gleichwertig zu benutzen, haben in der wissenschaftlichen Literatur zu geradezu absurden Verrenkungen geführt.[18] Für die folgende Arbeit wurde entschieden, zunächst von „freier Software“

---

[17] Bei Surfen im Web und Versenden von E-Mail müssen *Domain Name Server* abgefragt werden – auf 80 Prozent der 233 Millionen Domain-Server läuft dabei die freie Software BIND. Vgl. INTERNET SYSTEMS CONSORTIUM: ISC publishes New Domain Survey Report. Pressemitteilung vom 10.03.2004. http://www.isc.org/about/press/?pr=2004031000 [Aufgerufen am 09.08.2004].
Webseiten werden von einem *Webserver* ausgeliefert. In 67,7 Prozent der Fälle handelt es sich dabei um die freie Software Apache. Vgl. heise online: Weiter starkes Wachstum bei Anzahl der Websites. 05.08.2004, 10:23 Uhr. http://www.heise.de/newsticker/meldung/49785 [Aufgerufen am 06.08.2004].
Fast jede zweite E-Mail (genauer gesagt: 45 Prozent davon) wird beim Provider von freien *Mail-Servern* wie Sendmail, Postfix oder Exim weitertransportiert. Vgl. WHEELER, David A.: Why Open Source Software / Free Software (OSS/FS, FLOSS, or FOSS)? Look at the Numbers!
http://www.dwheeler.com/oss_fs_why.html [Aufgerufen am 25.01.2005].
Und es ist außerdem gut möglich, daß bereits der Router, der das Büro der Journalisten mit dem Internet verbindet, herstellerseitig ein verstecktes GNU/Linux-System enthält.

[18] Vgl. Kapitel 3.5.2 Allerdings böten die gleichen wissenschaftlichen Texte genug empirisches Material, um einmal das Verhältnis von Akronymen und Lesbarkeit kritisch zu überprüfen. Dies bliebe einer kognitionspsychologischen Arbeit vorbehalten.

zu sprechen, wenn damit Computerprogramme gemeint sind, die quelloffen[19] sind und unter einer freien Lizenz[20] verbreitet werden.[21]

Umstritten ist aus ganz ähnlichen Gründen auch der Begriff „Linux“. Da dieser Name lediglich den engsten Kern des Betriebssystems beschreibt, viele weitere essentielle Bestandteile aber aus Stallmans GNU-Projekt stammen, ist es korrekter, von GNU/Linux zu sprechen – auch wenn sich in der Berichterstattung die kürzere Bezeichnung durchgesetzt hat. Ich werde im folgenden den Begriff „GNU/Linux“ verwenden, außer es geht allein um den Kern des Systems oder es handelt sich um Eigennamen bzw. etablierte Wortzusammensetzungen.

Die Arbeit folgt der „alten“ Rechtschreibung.

Motivation

Zu meiner Motivation ist anzumerken, daß ich neben meiner beruflichen Tätigkeit als Journalist seit zwei Jahren einen sehr alten Laptop[22] zuhause unter anderem als Webserver betreibe. Ohne freie Software[23] wäre dieses private Projekt nicht umzusetzen gewesen: Die Lizenzkosten für entsprechende proprietäre Programme übersteigen mein Budget für eine solche Bastelei bei weitem; abgesehen davon wären sie auf dem genannten PC gar nicht lauffähig.[24] Die Rätsel aber, die zur Nutzung von GNU/Linux auf diesem nostalgischen Stück Elektroschrott zu lösen waren, haben mir möglicherweise eine Ahnung von dem vermittelt, was der Begriff „Hacking“ meinen könnte.

---

19 Vgl. den Anhang: „Das Schwein guckt ins Uhrwerk oder Was ist Quellcode?"

20 Als freie Lizenzen gelten hier die, die der „Free Software Definition“ genügen. Vgl. STALLMAN, Richard Matthew: Free Software Definition. In: Ders.: Free Software, Free Society. Selected Essays. Boston: GNU-Press, 2002, S. 41-44. Das sind auch alle Open-Source-Lizenzen, denn die „Open Source Definition“ der Open Source Initiative entstand auf dieser Basis und enthält im Kern die gleichen Anforderungen. Open-Source-Lizenzen werden darum umgekehrt auch von der Free Software Foundation als „free software licenses“ akzeptiert, auch wenn sie viele davon gleichzeitig als „GPL-incompatible“ einstuft. Vgl. PERENS, Bruce: The Open Source Definition. In: DIBONA, Chris; OCKMAN, Sam; STONE, Mark (Hrsg.): Open-Sources. Voices from the Open Source Revolution. Sebastopol: O'Reilly, 1999, S. 171-188 und: FREE SOFTWARE FOUNDATION: „Various Licenses and Comments about Them“. Online veröffentlicht unter: http://www.gnu.org/licenses/license-list.html [Aufgerufen am 31.01.2005].

21 Diese Gleichsetzung macht auch inhaltlich Sinn: „Open Source“ ist nach Definition seiner Erfinder lediglich ein anderer Name, ein „Marketing“-Begriff für freie Software – es ist daher durchaus möglich, Open Source als „freie Software“ zu bezeichnen. Es wäre jedoch nicht korrekt, im Umkehrschluß jede freie Software als Open Source zu bezeichnen, wie jeder weiß, der einmal mit Free-Software-Aktivist Stallman gesprochen hat. Es spricht also nichts dagegen, von freier Software als „freier Software“ zu sprechen, ganz ungeachtet dessen, daß wissenschaftliche Texte nicht dem Marketing dienen sollten. Vgl. Kapitel 3.5

22 486SX mit 25 MHz Takt.

23 Unter anderem mit dem Betriebssystem Debian GNU/Linux, dem Apache Webserver, dem Postfix Mailserver und einem MySQL Datenbankserver.

24 So beträgt der Preis für den vergleichbaren „Microsoft Small Business Server“ inklusive der nötigen Mindestmenge an Client-Lizenzen ca. 2700 Euro. Die Systemvoraussetzungen sind dabei zwar mit einem Pentium 166Mhz und 64 MB Arbeitsspeicher relativ moderat, liegen aber weit über dem, was mein Zielsystem zu bieten hat.

1 Die Hacker-Ethik

Es ist ein relativ unbekannter Aspekt der modernen Computertechnik, daß fast alle relevanten Programme des Internets (mit Ausnahme des Browsers) und ein wesentlicher Bestandteil der Softwareinstallationen großer Unternehmen und auch von Regierungen[25] auf einer Philosophie beruhen, die direkt auf die 1959 entstandene Hacker-Ethik zurückgeht. Wie in den meisten Fällen, in denen das Wort „Hacker“ heutzutage in seinem eigentlichen Wortsinn verwendet wird, ist auch an dieser Stelle darauf hinzuweisen, daß es hier nicht um Computer-Kriminalität geht, sondern um eine Selbstdefinition von Computerexperten, die bereits existierte, lange bevor die Presse das Wort „Hacker“ entdeckte.

1.1 Der Begriff "Hack"

Mit dem Wort „Hack“ bezeichnen die Studenten am Massachusetts Institut für Technology (MIT) in Cambridge seit den frühen 50er Jahren gelungene, anspruchsvolle Streiche, wie zum Beispiel das Verpacken der Universitätskuppel in Silberfolie.[26] Der Begriff ist möglicherweise verwandt mit dem deutschen „aushecken“ oder „Heckmeck“.

Um die Bezeichnung „Hack“ zu verdienen, muß die Tat eine gewisses Maß an Kreativität erkennen lassen; böswilliges oder rüdes Verhalten fällt nicht darunter.[27] Das Institut toleriert diese Streiche[28] und hat auf seiner Internetseite die Anforderungen für einen solchen Hack offiziell als „Hacker-Ethik“ präzisiert: Ein Hack dürfe nicht gefährlich sein, heißt es da, er dürfe nichts beschädigen, niemanden physisch, geistig oder emotional verletzen und er müsse lustig sein – zumindest für die meisten, die ihn erleben.[29]

Eine weitergehende Deutung erfuhr der Begriff „Hack“ als Teil des Jargons der Mitglieder des Modelleisenbahnclubs „Tech Model Railroad Club“ (TMRC).[30] Dieser Club

---

25 So ist die Vernetzung des Auswärtigen Amtes mit den Botschaften weltweit über Free Software bzw. Open-Source-Software realisiert. Vgl. ANDRESEN, Fred; WOLF, Ulrich: Tux inside. Biotop für Pinguine: Joschka Fischers Auswärtiges Amt. Linux-Magazin, Ausgabe 05/2003. München: Linux New Media AG, S. 72-74.
und GRUNWALD, Lukas: Virtuelle Verschlußsache. BSI-VPN SINA wird kommerziell verfügbar. iX Magazin für professionelle Informationstechnik. Ausgabe 09/2004. Hannover: Heise-Verlag, S. 95-97.

26 LEVY, Steven: Hackers. Heroes of the Computer Revolution. New York: Dell, 1984; Penguin, 2001, S. 23. Das Buch beruht auf Interviews mit mehr als 100 Zeitzeugen aus der Computerszene und ist für die geschichtlichen Darstellungen in der vorliegenden Arbeit eine wesentliche Grundlage.

27 WILLIAMS verweist in diesem Zusammenhang auf ein 1990 erschienenes Journal des MIT-Museums, in dem diese und andere Bedingungen eines „Hacks“ beschrieben sein sollen. Vgl. WILLIAMS, Sam: Free as in freedom. Richard Stallman's Crusade for Free Software. Sebastopol: O'Reilly, 2002, S. 203-208.

28 Das MIT veröffentlicht die gelungensten Hacks unter:
http://hacks.mit.edu/ [Aufgerufen am 14.08.2004].

29 Wörtlich heißt es: „According to the ‘hacker ethic,’ a hack must: be safe, not damage anything, not damage anyone, either physically, mentally or emotionally, be funny, at least to most of the people who experience it. There is no way of enforcing this code, but anything that directly contradicts it will probably not be considered a „hack“ by most of the MIT community.“ Vgl. MIT-Website:
http://hacks.mit.edu/Hacks/misc/ethics.html [Aufgerufen am 14.08.2004].

30 Vgl. hier und in der folgenden Darstellung des TMRC: LEVY, a.a.O., S. 14-152.

ist bis heute[31] eine studentische Arbeitsgemeinschaft am MIT. In dieser Gruppe drückte das Wort „Hack“ erstmals ein Werturteil über eine technische Lösung aus:

> “When a piece of equipment wasn’t working, it was “losing”; when a piece of equipment was ruined, it was ‘munged’ (Mash Until No Good); the two desks in the corner of the room were not called the office, but the ‘orifice’; one who insisted on studying for courses was a ‘tool’; garbage was called ‘cruft’; and a project undertaken or a product built not solely to fulfill some constructive goal, but with some wild pleasure taken in mere involvement, was called a ‘hack’.”[32]

1.2 Die ersten Hacker

Als im Frühjahr 1959 am MIT im Studiengang Elektrotechnik der erste Kurs im Programmieren gegeben wurde,[33] waren viele der Studenten Mitglieder der TMRC-Untergruppe „Signals and Power“. Das lag an einer Reihe von Übereinstimmungen zwischen den Anforderungen des Computerkurses und den Vorkenntnissen der Bastler: Ihre Erfahrungen mit den elektrischen Schaltkreisen der Modelleisenbahn hatten ihnen ein Vorverständnis der „Hardware“ von Computern vermittelt.[34] Durch das stundenlange Neuverdrahten und Überprüfen des Signalsystems waren sie außerdem an eine Tätigkeit gewöhnt, die dem Schreiben und dem „Debugging“ (der Fehlerkorrektur) von Programmcode ähnlich ist.

Mit den TMRC-Mitgliedern hielt die spielerische Art der „Hobbyists“, ihre Wertvorstellungen und ihr Jargon Einzug in dieses frühe Stadium der Informationstechnik – traf aber bald auf Widerstände: Die Computer dieser Zeit waren mehrere Millionen Dollar teure Anlagen, die einen ganzen Raum füllten und spezielle Klimaanlagen benötigten, damit ihre Vakuum-Röhren nicht durchbrannten. Angestellte kümmerten sich um die Prägung der Lochkarten mit Instruktionen, die stapelweise (batch-processing) Eingabe dieser Karten, die Ausgabe der Ergebnisse sowie die allgemeine Wartung der Maschinen.[35] Die strengen Regeln für den Zugang zu den damals am MIT vorhandenen IBM-Rechnern 704 und 709 veranlaßt LEVY zum Vergleich der Angestellten mit Priestern und Unterpriestern[36] – eine Terminologie, die Raymonds sehr viel späterem Vergleich der traditionellen Softwareentwicklung mit dem Bau einer Kathedrale ähnelt.[37] Ein kreativer Umgang mit den Maschinen, ein „Hacken“, war durch diese Bürokratie ausgeschlossen.

---

31 Vgl. dazu die Homepage des Clubs: http://tmrc.mit.edu/. Der TMTC hat außerdem eine eigene Erklärung zum Thema „Hacken“ veröffentlicht und zwar unter: http://tmrc.mit.edu/hackers-ref.html [Beide aufgerufen am 14.08.2004].

32 LEVY, a.a.O., S. 23.

33 Der Dozent war John McCarthy, der sein Feld „Artificial Intelligence“ nannte und später Leiter des 1970 gegründeten AI Labs wurde, des Forschungslabors für Künstliche Intelligenz am MIT. Vgl. ebd., S. 24f.

34 Die Elektronik der Modelleisenbahn war durchaus mit komplexeren Systemen vergleichbar. So waren die verwendeten Relais und Schalter dem MIT von der Telefongesellschaft Western Electric gespendet worden; ihr Zusammenwirken ähnelte dem Telefonsystem: Zum Öffnen und Schließen von Schaltkreisen wurden Kommandos über ein angeschlossenes Telefon eingewählt. Vgl. WILLIAMS, a.a.O., S. 204.

35 Vgl. LEVY, a.a.O., S. 19 und 26f.

36 Ebd., S. 27.

37 RAYMOND, Eric Steven: The Cathedral and the Bazaar. Musings on Linux and Open Source by an Accidental Revolutionary. Sebastopol: O'Reilly, 2001. RAYMOND unterscheidet mit den Begriffen „Kathedrale“ und „Basar“ herkömmliche Methoden der Softwareentwicklung vom Open-Source-Entwicklungsmodell.

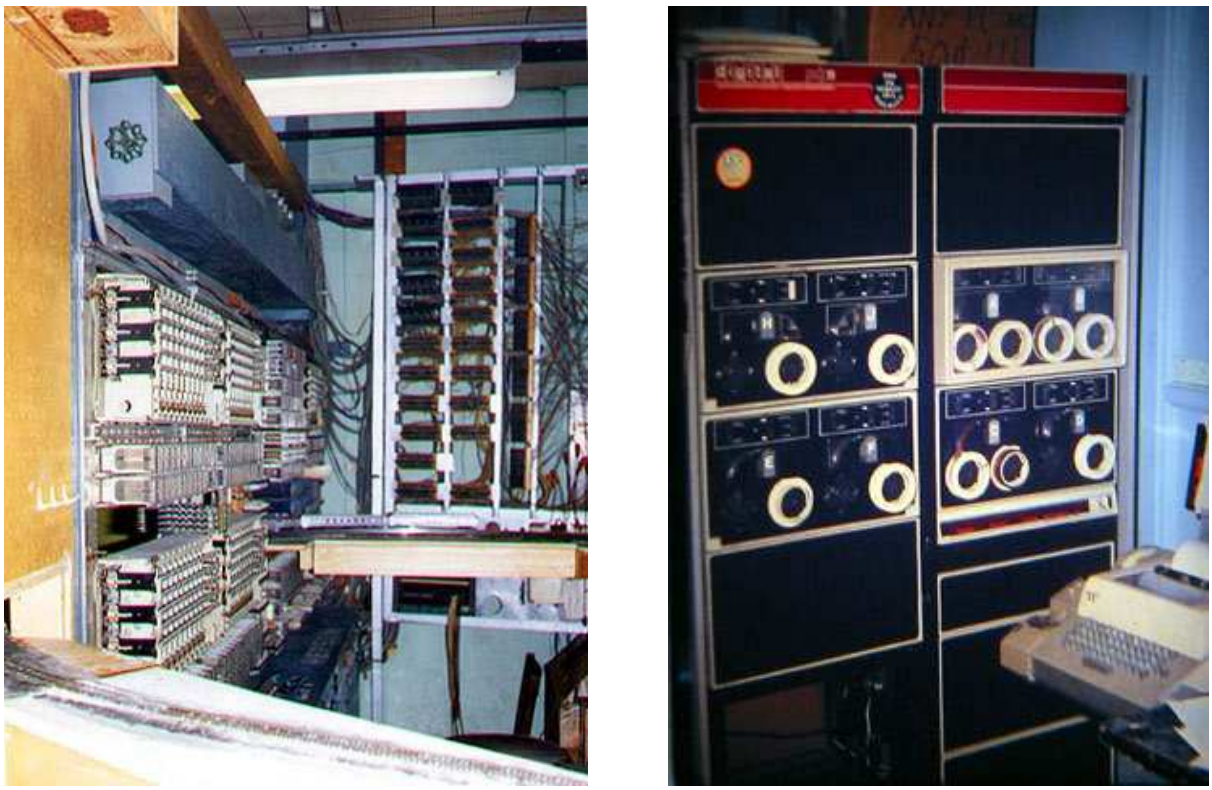


Abbildung 1: Die TMRC-Mitglieder bauten ihr Stellwerk aus Relais der Bell Telephone Company (linkes Bild). Der Club erhielt im Laufe der Zeit mehrere DEC- und Lincoln-Computer. Die rechts gezeigte PDP-11 des TMRC wurde lange Zeit zur Programmentwicklung genutzt.[38]

Angesichts dieser Situation beschlossen mehrere Professoren und Dozenten[39] – auch aus wissenschaftlichem Kalkül[40] – den interessierten Studenten vom TMRC mehr Freiheiten im Umgang mit Computern zu gewähren. Sie förderten und tolerierten es, daß diese Studenten in Zeiten außerhalb der regulären Inanspruchnahme einen TX-0-Computer (sprich: „Tixo“) benutzten; diesen drei Millionen Dollar teuren, aber bereits veralteten Großrechner hatte ihnen das „Lincoln Lab“, das militärische Entwicklungslabor am MIT, auf unbestimmte Zeit „ausgeliehen“.

An dem TX-0 und ab dem Jahr darauf auch an einem PDP-1-Minicomputer[41] entwickelte sich daraufhin eine Art Subkultur. Weil die Maschinen tagsüber für den regulären Universitätsbetrieb ausgelastet waren, lagen die Zugangszeiten für die TMRC-Mitglieder in der Nacht. Tagsüber wurden die Programme erdacht und aufgeschrieben, in den Club-

---

38 Quelle der Fotos: TMRC: A Brief History of the Tech Model Railroad Club. Fifty Years in Building 20. Online veröffentlicht unter: http://tmrc.mit.edu/history/ [Aufgerufen am 30.01.2005].

39 Es handelte sich vor allem um Jack DENNIS, John McKENZIE, John McCARTHY und Marvin MINSKY. Vgl. LEVY, a.a.O., S. 27ff und 66.

40 „McKenzie early on recognized, that the interactive nature of the TX-0 was inspiring a new form of computer programming and the hackers were its pioneers. So he did not lay down too many edicts.“ Ebd., S. 30.

41 Der PDP-1-Computer war einer der ersten, der einen Kathodenstrahl-Monitor als Ausgabemedium und ein Schreibmaschinen-Keyboard als Eingabemedium benutzte. Der Rechner wurde dem MIT 1960 von der Firma DEC geschenkt – DEC hoffte zu Recht, daß die Verwendung am MIT zu verwertbaren Programmen führen würde. Vgl. dazu: LISCHKA, Konrad: Als Computer Götter waren. „Spacewar!“ am MIT. In Telepolis – Magazin der Netzkultur, 04.08.2001.
http://www.heise.de/tp/deutsch/special/spiel/9223/1.html
Und: BELLIS, Mary: Inventors of the Modern Computer. Online veröffentlicht unter:
http://inventors.about.com/library/weekly/aa090198.htm [Beide aufgerufen am 06.09.2004].

räumen des TMRC diskutiert und (beim TX-0) mit Lochstreifen-Schreibern in eine computerlesbare Form gebracht.[42] Dieses Verfahren machte ein hohes Maß an Konzentration und Überblick beim Programmieren nötig – um dieses Niveau zu erreichen, lebten viele der Studenten einen 30-Stunden-Tag.[43] Weil die wenigen Computer-Arbeitsplätze so selten verfügbar waren, wurden die Rechner außerdem in Gruppen benutzt: Während die einen ihren Code eingaben, schauten andere zu und lernten. Dadurch wurde die eigene Software auch zu einer Form der Selbstdarstellung.[44]

Abbildung 2: PDP-1 Minicomputer der Fa. „Digital Equipment Corporation" (DEC) von 1961 [45]

Lange vor der Anonymität des Internet zeigten sich bereits in dieser räumlich beengten Form der Zusammenarbeit Anzeichen einer Meritokratie, also einer idealisierten Organisationsform, in der die Höhe der Wertschätzung nur vom Ausmaß der eingebrachten intellektuellen Fähigkeiten abhängig ist und nicht von äußerlichen Eigenschaften wie Geschlecht, Hautfarbe oder Alter: Bereits im Jahr 1959 wurde Peter Deutsch aufgrund seiner

---

42 Vgl. LEVY, a.a.O., S. 35f.

43 So schreibt LEVY etwa über die Arbeitsweise des Programmierers Richard Greenblatt im Jahr 1963: „He was turning out an incredible amount of code, hacking as much as he could, or sitting with a stack of printouts, marking them up. He'd shuttle between the PDP-1 and TMRC, with his head fantastically wired with the structures of the program he was working on, or the system of relays he'd hacked under the TMRC layout. To hold that concentration for a long period of time, he lived, as did several of his peers, the thirty-hour day. It was conducive to intense hacking, since you had an extended block of waking hours to get going on a program, and, once you were really rolling, little annoyances like sleep need not bother you. The idea was to burn away for thirty hours, reach total exhaustion, then go home and collapse for twelve hours. An alternative would be to collapse right there in the lab." Ebd., S. 74.

44 Vgl. Ebd., S. 118 sowie S. 78. Daß Programmieren auch eine Form der Selbstdarstellung ist, ist außerdem Teil der Hacker-Ethik, die im späteren Verlauf dieser Arbeit beschrieben wird.

45 Quelle: Columbia University, New York. Online veröffentlicht unter:
http://www.columbia.edu/acis/history/pdp-1.html
[Aufgerufen am 31.01.2005].

Fähigkeiten an dem TX-0-Computer von den TMRC-Mitgliedern als gleichrangig akzeptiert, obwohl der Sohn eines MIT-Professors erst 12 Jahre alt war.[46]

Diese erste „Community" in der Geschichte der Softwareentwicklung stieß nicht bei allen MIT-Offiziellen auf Wertschätzung. So beschreibt Joseph WEIZENBAUM, seit 1963 einer der Informatik-Professoren am MIT, die Computer-Enthusiasten in deutlich negativen Worten:

> "Wherever computer centers have become established, that is to say, in countless places in the United States, as well as in virtually all other industrial regions of the world, bright young men of disheveled appearance, often with sunken glowing eyes, can be seen sitting at computer consoles, their arms tensed waiting to fire, their fingers, already poised to strike at the buttons and keys on which their attention seems to be as riveted as a gambler's on the rolling dice. When not so transfixed, they often sit at tables strewn with computer printouts over which they pour like possessed students of a cabalistic text. They work until they drop, twenty, thirty hours at a time. Their food, if they arrange it, is brought to them: coffee, cokes, sandwiches. If possible they sleep on cots near the computer. But only for a few hours – then back to the console or the printouts. Their rumpled clothes, their unwashed and unshaven faces, and their uncombed hair all testify that they are oblivious to their bodies and to the world in which they move. They exist, at least when so engaged, only through and for computers. These are computer bums, compulsive programmers. They are an international phenomenon."[47]

WEIZENBAUM bestritt zwar bereits damals, mit dieser Beschreibung die Studenten am MIT portraitiert zu haben und verwahrte sich im Jahr 2005 erneut gegen diesen Vorwurf (siehe Interview im Anhang). Dennoch fühlten sich damals einige der MIT Hacker, unter ihnen auch der später wegen seiner Programmierkünste bekanntgewordene Richard Greenblatt, in einem solchen Ausmaß persönlich verunglimpft, daß sie schriftlich gegen die Darstellung protestierten.[48]

Ein anderer MIT-Professor, Marvin MINSKY – später WEIZENBAUMs Hauptgegner in der Diskussion um die moralischen Folgen der Informatik – nahm die Studenten ausdrücklich gegen ihn in Schutz:

> "'He's just crazy,' Minsky would later say of Weizenbaum. 'These hackers are the most sensitive, honorable people that ever lived.' Hyperbole perhaps, but it was true that behind their single-mindedness there *was* warmth, in the collective realization of the Hacker Ethic."[49]

---

[46] Ebd., S. 30f. Wobei sich bei diesem von Levy gezeichnete Idealbild die Frage aufdrängt, warum zwar 12jährige Jungen unter den TMRC-Hackern als gleichrangig galten, es in dieser Gruppe aber überhaupt keine weiblichen Studenten gab.

[47] WEIZENBAUM, Joseph: Computer Power and Human Reason: From Judgement to Calculation. San Francisco 1976. Zitiert nach: LEVY, a.a.O., S.133f. Das Buch erschien zu deutsch als: WEIZENBAUM, Joseph: Die Macht der Computer und die Ohnmacht der Vernunft. Frankfurt am Main: Suhrkamp, 1978. Das Wort „computer bum" ist eine eigene Erfindung Weizenbaums, er mag es von „beach bum" abgeleitet haben, was soviel bedeutet wie „Strandgammler".

[48] Vgl. LEVY, a.a.O., S.134.

[49] Ebd., S.138.

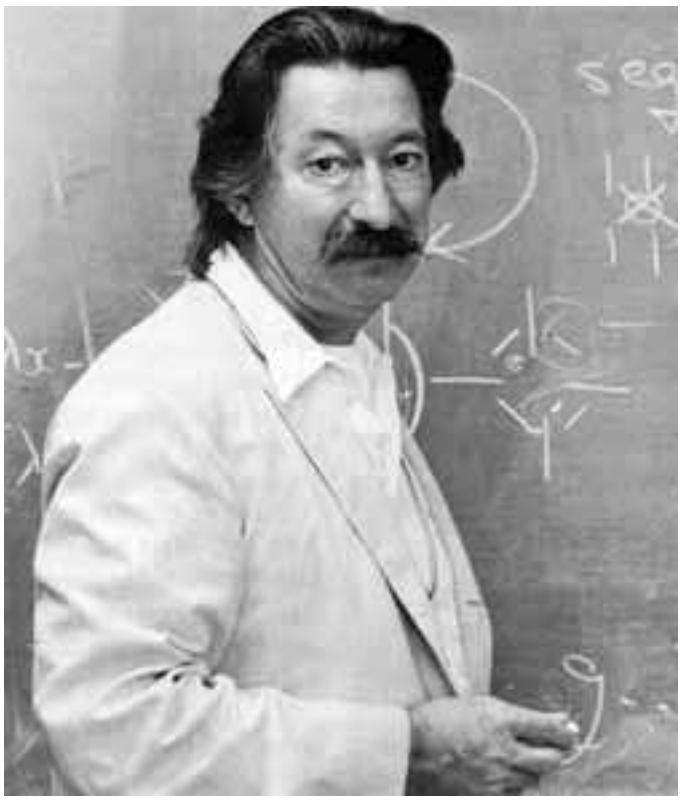


Abbildung 3: MIT-Professor Joseph Weizenbaum im Jahr 1975[50]

Die unbegrenzten Möglichkeiten des Computers erfuhren die ersten Hacker unter den Bedingungen starker zeitlicher und räumlicher Begrenzungen. Aus dieser Erfahrung heraus schufen sich die vermeintlich unsozialen ersten Programmierer einen eigenen Ehrenkodex, der die Werte des Modelleisenbahn-Clubs und seiner kreativen Hobby-Bastelei mit den Werten des MIT und seiner ambitionierten Streich-Kultur verband: Die Hacker-Ethik.

1.3 Die Regeln der Hacker-Ethik

Der amerikanische Wissenschaftsjournalist Steven LEVY hat aus seinen Interviews mit den ersten Hackern in Boston und in Kalifornien sechs Regeln der sogenannten Hacker-Ethik entwickelt, die im folgenden mit einer Übersetzung ins Deutsche aufgelistet werden:

1. "Access to computers – and anything which may teach you something about the way the world works – should be unlimited and total. Always yield to the Hands-On Imperative!"[51]
   „Der Zugang zu Computern – wie zu allem, das einem etwas darüber beibringen kann, wie die Welt funktioniert – sollte vollständig und unbegrenzt sein. Vorrang hat immer das Recht, die Dinge selbst in die Hand zu nehmen."
2. "All information should be free."[52] – „Alle Information muss frei sein."

---

50 Das Bild von Joseph Weizenbaum stammt aus: MIT: Laboratory for Computer Science (formerly Project Mac), Massachusetts Institute of Technology. 1975. Veröffentlicht online unter: http://medg.lcs.mit.edu/people/psz/LCS-75/Brochure.html [Aufgerufen am 10.10.2005].

51 Ebd., S.40.

52 LEVY, a.a.O., S.40.

3. “Mistrust authority – Promote Decentralization.”[53]
   “Mißtraue Autoritäten – kämpfe für dezentrale Lösungen.“
4. “Hackers should be judged by their hacking, not bogus criteria such as degrees, age, race, or position.”[54]
   “Hacker sollten nach ihrer Leistung im Hacken beurteilt werden, nicht nach fadenscheinigen Kriterien wie Abschluss, Alter, Hautfarbe oder Rang.“
5. “You can create art and beauty on a computer.”[55]
   “Du kannst Kunst und Schönheit an einem Computer erschaffen.“
6. “Computers can change your life for the better.”[56]
   “Computer können Dein Leben zum Besseren wenden.“

Diese Regeln sollen im folgenden beschrieben und anhand von weiterer Literatur über die Hacker-Kultur diskutiert werden.

1.3.1 Freier Zugang zu Computern

Die erste Regel der von LEVY beschriebenen Hacker-Ethik lautet: “Access to computers – and anything which may teach you something about the way the world works – should be unlimited and total. Always yield to the Hands-On Imperative!”[57] Für diesen Anspruch auf unbeschränkten Zugang zu Computern[58] (und Computernetzen) lassen sich zunächst situative Erklärungen finden: Historisch kann man ihn aus dem Mangel an Computer-Arbeitsplätzen erklären, mit dem die ersten Hacker zu kämpfen hatten. Die Regel mag ihre Wurzeln außerdem in der studentischen Tradition des sogenannten „lock-hacking“ am MIT gründen, zu deutsch: des Umgehens von Schlössern.[59]

LEVY sieht diese Forderung der Hacker aber noch grundsätzlicher: Die „Hands-On Imperative“[60], also der Anspruch darauf, die Dinge selbst in die Hand zu nehmen und bei Bedarf zu ändern, sei für Hacker nicht nur ein natürlicher Ausdruck von Forscher- und Bastlergeist, sondern notwendiger Anspruch eines mündigen Menschen.

Dieser sei keineswegs auf die Welt der Computer reduziert, schreibt LEVY und nennt als Beispiel den Ärger über eine schlecht geschaltete Verkehrsampel:

---

53 LEVY, a.a.O., S.41.

54 LEVY, a.a.O., S.43.

55 LEVY, a.a.O., S.43.

56 Ebd., S.45.

57 Ebd., S.40.

58 Meiner Auffassung nach dient der Zusatz „and anything which may teach you something about the way the world works“ LEVY lediglich dazu, das Hacker-Motiv des *Erforschens* zu betonen. Es ist nicht ersichtlich – weder aus LEVYs Beschreibung noch aus anderen Betrachtungen der Hacker-Kultur – daß von Hackern jemals ein ernsthafter Anspruch auf den freien Zugang zu anderen Gütern außerhalb des Bereichs der Informationstechnik erhoben oder durchgesetzt wurde.

59 Dies ist völlig physisch und keineswegs virtuell gemeint: Es ging den Studenten damit sogar um den Zugang zu den Büros ihrer Dozenten – diese Art Streich wurde auch von den frühen Programmierern lebendig gehalten. Vgl. LEVY, a.a.O., S.102-107.

60 LEVY bezeichnet das Bedürfnis von Bastlern, ein defektes oder zu verbesserndes Geräteteil selbst in die Hand zu nehmen, als „Hands-On Imperative“. Schon um Fehlermöglichkeiten auszuschließen, sei für sie der eigene Augenschein grundsätzlich den Erfahrungen aus zweiter Hand vorzuziehen. Vgl. ebd., S.22.

> In a perfect hacker world, anyone pissed off enough to open up a control box near a traffic light and take it apart to make it work better should be perfectly welcome to make the attempt. Rules which prevent you from taking matters like that into your own hands are too ridiculous to even consider abiding by. [61]

Daß der Anspruch der Hacker, schlecht funktionierende Dinge selbst zu ändern, noch nicht einmal auf die Welt der Technik beschränkt ist, zeigt sich im Fall von STALLMANS GNU General Public License: Aus Unzufriedenheit mit den Beschränkungen des Copyright schuf STALLMAN eine neue Vertragsform auf Basis des Urheberrechts und befreite damit die Nutzer von diesen Beschränkungen – eine Leistung, die auch als „Copyright Hack" beschrieben wird und das Urheberrecht in der Tat verändert hat.[62]

Fast eine Definition dieser ersten Regel der Hacker-Ethik und ihres gesellschaftlichen Anspruchs liefert MAUS in seiner Reportage über ein deutsches Hacker-Camp – immerhin gut zwanzig Jahre nach LEVYs Interviews:

> „Es geht darum, die immer komplizierteren Technologien auseinanderzunehmen, hineinzuschauen und wieder neu zusammenzusetzen. Das spielerische Basteln hat einen aufklärerischen Impetus. Wer die Technik nicht beherrscht, wird von ihr beherrscht. Hier ist Kant der Geist in der Maschine. Das Aufbrechen geschlossener Systeme ist der Weg aus der technischen Unmündigkeit. Der offenen Gesellschaft stehen offene Systeme zu. Hier kämpft jeder für die Open-Source-Bewegung. Jede Software unter Open-Source-Lizenz legt ihren Programmcode offen. Technik darf keine Blackbox mit Hintertürchen sein. Es geht nicht nur um Marktmonopole, sondern auch darum, Wissensmonopole zu knacken."[63]

Bemerkenswert an dieser ersten Regel ist auch, daß sie von einem grundsätzlichen Altruismus aller Hacker ausgeht: Es wird nämlich nicht nur für jeden ein freier Zugang zu und ein freier Umgang mit Computern (und anderen Dingen) gefordert, sondern auch unterstellt, daß niemand diesen Zugang zu Lasten anderer Hacker ausnutzt.

1.3.2 Informationsfreiheit

„All information should be free",[64] diese Forderung nach Informationsfreiheit ist wohl die meistzitierte Regel der Hacker-Ethik. LEVY bezieht dabei den Begriff „Information" zunächst nur auf Quellcode. Dessen völlige Freilegung beschreibt er als eine Art Überlebensstrategie der ersten Hacker: Weil die frühen Computer nur mit sehr dürftiger Software (wenn überhaupt) ausgeliefert wurden, bestand die erste Aufgabe der Anwender darin, sich Programmierwerkzeuge zur Benutzung der Maschine zu erstellen, z. B. Systemprogramme und Editoren. LEVY nennt diese Programme "Tools to make tools". Damit niemand das Rad von vorne erfinden mußte, wurden diese Programme (als Magnetbänder) am MIT grundsätzlich für alle zugänglich in Schubfächern in der Nähe der jeweiligen

---

61 Ebd., S.40.

62 Vgl. Kapitel 2.3.3 „Die GNU General Public License (GPL)".

63 MAUS, Stephan: Ins Biotop der Hacker-Gemeinde. Chaos, Computer und Club: Eine Existenz zwischen Breitmaulfrosch, Klapperstorch und dem Geist der Maschine. Süddeutsche Zeitung, 12.08.03, S. 13 (Feuilleton).

64 LEVY, a.a.O., S.40.

Computer aufbewahrt.[65] Später ersetzten anonyme FTP-Server bzw. offene Systemverzeichnisse die Schubfächer.

Außerdem waren zu der Zeit der ersten Hacker Anwender und Programmierer noch identisch. Wer damals an Computern arbeitete, war es gewohnt, Probleme mit Druckern zu lösen, indem man selbst den Treiber umschrieb und anpaßte. Ohne freien Zugriff auf Quellcode hätte es in der Startphase des Computerzeitalters keinen (oder nur viel langsameren) Fortschritt gegeben.

Neben diesen historischen Zwängen sieht LEVY aber auch eine weitere Wurzel für die Regel „All information should be free" – nämlich die Maschinen selbst: Hacker, so glaubt er, betrachteten die generelle Funktionsweise von Computern und Computerprogrammen als Leitbild für ihren eigenen Umgang mit Informationen.

> "The belief, sometimes taken unconditionally, that information should be free was a direct tribute to the way a splendid computer or computer program, works – the binary bits moving in the most straightforward, logical path necessary to do their complex job. What was a computer but something which benefited from a free flow of information? [...] In the hacker viewpoint, any system could benefit from that easy flow of information." [66]

Die Grundregel, daß Information frei sein soll, wird auch zitiert als „Information ***wants*** to be free" – eine sprachliche Veränderung, die die Regel nahezu in den Rang eines Naturgesetzes stellt. Als Urheber dieses Ausdrucks gilt Stewart BRAND, der das Streben von Information nach Freiheit als natürliches Ergebnis zweier zunächst paradox erscheinenden Eigenschaften von Informationen sah – sie sind sowohl billig in der Verbreitung als auch teuer im Wert: „Information wants to be free because it has become so cheap to distribute, copy, and recombine – too cheap to meter. It wants to be expensive because it can be immeasurably valuable to the recipient. That tension will not go away."[67]

Der amerikanische Dichter John Perry BARLOW macht diese Naturgesetzlichkeit noch deutlicher, indem er Information schlechthin zu einer Lebensform erklärt, zu einer Art Parasit im Wirtskörper Mensch:

> "I believe they are life forms in every respect but their freedom from the carbon atom. They self-reproduce, they interact with their surroundings and adapt to them, they mutate, they persist. They evolve to fill the empty niches of their local environments, which are, in this case the surround-ing belief systems and cultures of their hosts, namely, us. [...] Like DNA helices, ideas are relentless expansionists, always seeking new opportunities for Lebensraum. And, as in carbon-based nature, the more robust organisms are extremely adept at finding new places to live. Thus, just as the common housefly has insinuated itself into practically every ecosystem on the planet, so has the meme of "life after death" found a niche in most minds, or psycho-ecologies." [68]

Diese etwas esoterische Betrachtungsweise BARLOWS, Songschreiber der Band „The Grateful Dead" und Verfasser der „Unabhängigkeitserklärung des Cyberspace"[69] wird von

---

65 Ebd., S.41.

66 Ebd., S.41.

67 BRAND, Stewart: The Media Lab: Inventing the Future at MIT. New York: Viking Penguin, 1987. S.202.

68 BARLOW, John Perry: The Economy of Ideas. In: Wired, Ausgabe 2/03, März 1994. http://www.wired.com/wired/archive/2.03/economy.ideas.html [Aufgerufen am 15.06.2004].

69 BARLOW, John Perry: A Declaration of the Independence of Cyberspace. Online veröffentlicht unter: http://www.eff.org/~barlow/Declaration-Final.html [Aufgerufen am 25.10.2004].

Unix-Programmierern sogar noch auf die Spitze getrieben. Ihrem Verständnis nach ist Information nur lebendig (und frei), wenn sie in lesbarer Form in Computern (und Netzwerken) vorliegt – und nicht etwa ausgedruckt auf Papier:

> "Once you have printed your data on paper, you have largely lost the ability to manipulate it any furher. Data on paper cannot be sorted, moved, filtered, transformed, modified, or anything else as easily as it can on a computer. You cannot search it at the touch of a key. [...] It's simply not possible to move paper data as fast as electronic bits stored on a computer. Therefore, paper data will always be 'stale' compared with data kept on a computer. [...] Beware of paper. It is a death certificate for your data."[70]

Letztlich bleibt die Hacker-Regel „All information should be free" mehrdeutig. Sie kann verstanden werden

- als Aufruf zum kooperativen Umgang mit Informationen und zur Vermeidung von Herrschaftswissen,
- als Beschwörung eines freien Kommunikationsflusses analog zur digitalen Welt,
- als ökonomische Erkenntnis, daß Informationen nicht mehr an die Verbreitung physischer Datenträger gebunden sind oder
- als Propagierung einer informationellen Gratiskultur.[71]

Möglicherweise liegt es an dieser Mehrdeutigkeit, daß diese Regel auch eine starke emotionale Bindung entfaltet. Den missionarischen Eifer, mit dem sich Hacker an die Entdeckung und elektronische Verbreitung von neuem, verborgenen oder verbotenem Wissen machen, beschreibt STERLING als den aufziehenden Geist der Informationsgesellschaft:

> "However, hackers are *very* serious about forbidden knowledge. They are possessed not merely by curiosity but by a positive *lust to know*. The desire to know what others don't is scarcely new. But the *intensity* of this desire, as manifested by these young technophilic denizens of the Information Age, may in fact *be* new and may represent some basic shift in social values – a harbinger of what the world may come to, as society lays more and more value on the possession, assimilation, and retailing of *information* as a basic commodity of daily life."[72]

1.3.3 Mißtrauen gegen Autorität

„Mistrust authority – Promote Decentralization."[73] lautet die dritte Regel der Hacker-Ethik. Auch diese Regel führt LEVY auf die Erfahrungen der frühen Hacker zurück. Ihre

---

[70] GANCARZ, Mike: Linux and the Unix Philosophy. New York: Elsevier Science, 2003, S.115. Es handelt sich bei diesem Buch um eine Einführung in Überzeugungen und grundlegende Modelle von Unix-Programmierern. Interessant ist m. E., daß hier Informationen und Daten weitgehend als Rohstoff angesehen werden; die Begriffe „Freiheit" bzw. „Lebendigkeit" bezeichnet nicht mehr als den Aggregatzustand dieses Rohstoffes ("kopierbar" bzw. „computerlesbar"). Es zeigt sich ein großer Unterschied, vergleicht man diese Definition mit dem Verständnis, das die ersten Buchdrucker oder die Verfasser der ersten Flugblätter von „freier Information" oder „lebendigen Worten" hatten.

[71] Daß es ihnen um eine informationelle Gratiskultur gehen könnte, würde freilich von Anhängern der Open-Source- und der Free-Software-Bewegung energisch bestritten – schließlich ist es das Hauptvorurteil, gegen das diese Bewegungen seit ihrem Bestehen zu kämpfen haben.

[72] STERLING, Bruce: The Hacker Crackdown. Law and Disorder On The Electronic Frontier. New York: Bantam Books, 1993, S. 78f.

[73] LEVY, a.a.O., S.41.

Aversion gegenüber Autoritäten hätten die Hacker am MIT zu einer Zeit entwickelt, als sie sich den engen Vorschriften ausgeliefert sahen, die für den Zugang zu den IBM-Rechnern 704 und 709 galten. Dieses System aus Antragsformularen, Wartezeiten, Zugriffsbeschränkungen und Privilegien sei das Gegenteil der informellen Atmosphäre gewesen, die sich später entwickelte, als die Bastler einen großzügigen Zugang zu TX-0 und PDP-1 erhalten hatten. Autoritäten und Bürokratien hatten sie als Haupthindernis für die Erfüllung der Regeln eins und zwei erfahren: Sie regelten sowohl den Zugang zu Computern als auch den zu Informationen nach einer Logik, die nicht sach- sondern machtorientiert war – und damit das Gegenteil einer Computerlogik.

> "Bureaucracies, whether corporate, government, or university, are flawed systems, dangerous in that they cannot accommodate the exploratory impulse of true hackers. [...] Bureaucrats hide behind arbitrary rules (as opposed to the logical algorithms by which machines and computer programs operate): they invoke those rules to consolidate power, and perceive the constructive impulse of hackers as a threat." [74]

Bürokratien und Autoritäten werden aber nicht nur rational als sachlich schlechtere Lösung im Umgang mit Computern verworfen. Die Hacker lehnen sie auch aus emotionalen Gründen ab. LEVY sieht diese Ablehnung in der Kindheit und Jugend der Technikfans begründet. Unter ihren Altersgenossen seien sie Außenseiter gewesen; ihre Fähigkeiten an Computern seien ihre wichtigste Quelle von Machtgefühlen: „So it was natural to distrust any force which might try to limit the extent of that power."[75]

In ähnlicher Weise argumentiert Bruce STERLING: Er führt die Ablehnung von Autorität auf ein narzißtisches, romantisierendes Selbstbild des Hackers zurück – eine Selbstsicht als „techno-cowboy"[76], als Teil einer Elite, die über den unwichtigen Regeln von Telekommunikationsfirmen und inkompetenten Politikern steht.[77]

> "Hackers of all kinds are absolutely soaked through with heroic anti-bureaucratic sentiment. Hackers long for recognition as a praiseworthy cultural archetype, the postmodern electronic equivalent of the cowboy and mountain man." [78]

Es ist diese rebellische, antiautoritäre Haltung, die das Bild von Hackern in der Öffentlichkeit noch immer maßgeblich bestimmt. Seit Mitte der 80er ist das Wort ein Synonym für einen „Computerkriminellen". Wie STERLING ausführt, liegt dies nicht zuletzt an den damaligen Crackern und Telefon-Freaks (sog. „Phreaks") selbst, die ihre zweifelhaften Machenschaften, etwa den Diebstahl von Kreditkartennummern oder von Prepaid-Telefoncodes als „Hacking" bezeichneten und sich selbst als „Hacker".[79]

---

74 Ebd.

75 Ebd., S.43. Und möglicherweise waren die ersten Adressaten dieses Mißtrauens die Eltern der Hacker.

76 STERLING, a.a.O., S. 51.

77 Ebd., S. 55f.

78 Ebd., S. 51.

79 Vgl. ebd., S. 53.

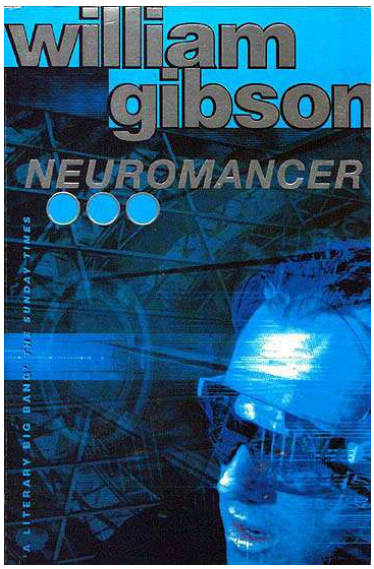


Abbildung 4: Das Selbstbild des Hackers als „Techno-Cowboy“ spiegelt sich wieder im Science-Fiction-Genre des „Cyberpunk“, allen voran in William Gibsons 1984 erschienenem Roman „Neuromancer“.[80]

Schon das ursprüngliche Verständnis eines „Hacks“ am MIT als eine Art intellektuellen Streichs schloß eine Mißachtung von Regeln und Autoritäten ein. Dies wurde auch von den Computerstudenten am MIT so verstanden. Bereits im Winter 1960 nutzte etwa der Student Alan Kotok Schwächen des Campus-Telefonsystems, um kostenlose Telefonate zu führen.[81] Ein weiterer Student, Stewart Nelson, dehnte diese Experimente Mitte der 60er auf das gesamte Bell-Telefonsystem aus.[82] LEVY zufolge folgten die Studenten dabei aber immer der inoffiziellen Hacker-Moral ihrer Gemeinschaft: “You could call anywhere, try anything, experiment endlessly, but you should not do it for financial gain.”[83] Neben LEVY und STERLING[84] betont auch Raymond, daß Hacker selbst jede Art krimineller Machenschaft nicht nur als unwürdig, sondern auch als unfähig ansehen:

> “Real hackers call these people ‘crackers’ and want nothing to do with them. Real hackers mostly think crackers are lazy, irresponsible, and not very bright, and object that being able to break security doesn’t make you a hacker any more than being able to hotwire cars makes you an automotive engineer. [...] The basic difference is this: hackers build things, crackers break them.”[85]

Statt mit Kriminellen vergleicht GRAHAM die antiautoritäre Haltung der Hacker vielmehr mit einer ähnlichen Attitüde des amerikanischen Mathematikers und Atomphysikers Richard Phillips Feynman: Während er am streng-geheimen Manhattan-Projekt arbeitete, amüsierte er sich damit, in Safes mit Geheimdokumenten einzubrechen.[86] GRAHAM

---

80 Quelle: GIBSON, William: Neuromancer. Pymble (Australia): Voyager, 1995. Titelbild.

81 LEVY, a.a.O., S.51.

82 Ebd., S.94f.

83 Ebd., S.94.

84 Vgl. etwa STERLING, a.a.O., S. 94.

85 RAYMOND, Eric Steven: How to Become a Hacker. In: Ders.: The Cathedral and the Bazaar. Musings on Linux and Open Source by an Accidental Revolutionary. Sebastopol: O'Reilly, 2001, S. 196.

86 Vgl. GRAHAM, Paul: Hackers & Painters. Big Ideas from the Computer Age. Sebastopol: O'Reilly, 2004, S. 50. Der spätere Nobelpreisträger wies die verantwortlichen US-Militärs anschließend auf diese Unsicherheit hin. Diese reagierten, indem sie nicht die Safes besser sicherten, sondern Feynman den Zugang zu ihnen verboten.

sieht dies als Ausdruck einer intellektuelle Neugier, die auch die Qualität eines Programmierer ausmache: „Those in authority tend to be annoyed by hackers' general attitude of disobedience. But that disobedience is a byproduct of the qualities that make them good programmers."[87]

Die Regel „Mistrust authority – Promote Decentralization" ist somit als eine Absage an Denkverbote zu verstehen, der Hacker erklärt sich mit ihr zum unabhängigen Geist. Nicht unbedeutend ist aber auch ihre wörtliche Bedeutung als Faustregel für die Organisation der Hacker-Community. Wie im weiteren Verlauf dieser Arbeit zu sehen sein wird, spielen Dezentralisierung und der zurückhaltende Gebrauch von Autorität bei der Linux-Entwicklung eine tragende Rolle.[88]

1.3.4 Meritokratie

Als vierte Regel der Hacker-Ethik benennt LEVY folgende Überzeugung der von ihm beobachteten Hacker-Kultur: „Hackers should be judged by their hacking, not bogus criteria such as degrees, age, race, or position."[89]

LEVY verweist im Zusammenhang mit dieser Regel auf den bereits erwähnten zwölfjährigen Peter Deutsch, der von den Hackern am MIT wegen seiner Programmierfähigkeiten als gleichgestellt betrachtet wurde, während andere trotz nachgewiesenem formalen Ausbildungsstand nicht akzeptiert wurden, wenn ihre tatsächlichen Kenntnisse an der Konsole nicht ausreichten.[90]

Reputation und Status erhält ein Hacker nach seinen Leistungen als solcher und nicht nach fachfremden Gesichtspunkten – mit dieser Regel definiert sich diese Gemeinschaft also als Meritokratie. Obwohl damit auch ein Diskriminierungsverbot einhergeht, ist diese Regel nicht notwendigerweise aus sozialen Gründen entstanden: Sie spiegelt vor allem die logische Sichtweise von Hackern: In einer Welt des puren Intellekts zählt nur die intellektuelle Leistung. "This meritocratic trait was not necessarily rooted in the inherent goodness of hacker hearts – it was mainly that hackers cared less about someone's superficial characteristics than they did about his potential to advance the general state of hacking."[91]

„Im Internet weiß niemand, daß du ein Hund bist" sagt ein Hund zum anderen auf einer inzwischen klassischen Karikatur der Zeitschrift „New Yorker". Dies gilt nicht nur für das Internet sondern auch für Hacker-Communities und ihren heutigen Pendants, den freien Softwareprojekten. Die an den Existenzialismus erinnernde Philosophie, nach der nur die Taten als Hacker zählen und keine Attribute wie Abschlüsse, Alter, Hautfarbe oder gesellschaftliche Position, mag der Intellektuelle oder „Nerd" als ebenso befreiend erleben wie der abgebildete Hund.[92]

---

87 Ebd., S. 51

88 Siehe Kapitel 3.2.2.

89 LEVY, a.a.O., S.43.

90 Ebd.

91 Ebd.

92 Die Anonymität und die aus ihr resultierende „objektive" Beurteilung von Leistungen mag auch die Anziehungskraft dieser virtuellen Gemeinschaften erklären. Die Unabhängigkeit von den genannten „unerheblichen Kriterien" ist möglicherweise schon immer ein wichtiges Motiv für die Verwendung der Schriftsprache gewesen – mit gelegentlich tragischen Folgen, portraitiert etwa im unglücklichen Cyrano de Bergerac.

Abbildung 5: „Im Internet weiß niemand, daß du ein Hund bist". The New Yorker, 1993[93]

1.3.5 Programmieren als Kunst

Die fünfte Regel der Hacker-Ethik formuliert LEVY als: „You can create art and beauty on a computer."[94] Diese Regel hat aber nichts mit Zeichen- oder Musikprogrammen zu tun. Kunst und Schönheit betreffen vielmehr die eigentliche Tätigkeit des Hackers: ***Programmieren***.

LEVY zufolge galt es 1960 als besonders ästhetischer Programmierstil, wenn es gelang, komplizierte Aufgaben mit möglichst wenigen Instruktionen zu erreichen.[95] Schon aufgrund des geringen Arbeitsspeichers der historischen Computer waren die Programmierer zu Sparsamkeit gezwungen. Das an das Verfassen eines Haikus erinnernde „program bumming"[96], also das Komprimieren eines Programms auf möglichst wenige Befehle, erforderte außerdem höhere Kenntnisse und auch ungewöhnliche Ideen des Programmierers: So kann die Zahl der Zeilen eines Programms etwa durch die Nutzung von Variablen, von Loops (Zirkelbefehlen) oder Algorithmen reduziert werden – wenn der Programmierer selbst zu den nötigen Abstraktionen fähig ist.

Wie beschrieben lagen die Programme dieser Hackergemeinschaft am MIT grundsätzlich im Quellcode vor. Aufgrund der Knappheit an Computerterminals beobachtete man sich gegenseitig beim Eingeben von Code und lernte daraus. Aus diesen beiden Gründen blieben die verwendeten Tricks und minimalistischen Techniken kein Geheimnis und

---

93 STEINER, Peter: Karikatur. In: New Yorker, 05.07.1993, S. 61. Siehe auch: http://www.unc.edu/depts/jomc/academics/dri/idog.html [Aufgerufen am 29.10.2004].

94 LEVY, a.a.O., S.43.

95 Vgl. ebd.

96 Das schöne englische Wort „bum", das sowohl „Gesäß" als auch „Gammler" bedeuten kann, findet hier (nach WEIZENBAUMS Beurteilung der ersten Hacker) erneut Eingang in die Computersprache. Diesmal soll es MIT-Computerprofessor John McCarthy gewesen sein, der die Leidenschaft der Studenten, Programmcode zu verdichten mit dem Enthusiasmus von „ski-bums", also fanatischen Skiläufern, verglich. Vgl. LEVY, a.a.O., S.26.

stießen in der Hacker-Gemeinschaft auf hohe Anerkennung, bis hin zu Verehrung. LEVY bezeichnet sie als „genius-from-Mars Technique“, als „black magic“, als „magical stunts“, den mühsamen Weg dorthin vergleicht er mit der Suche („quest“) nach dem Heiligen Gral.[97]

Code wurde zur Quelle von Anerkennung und zum Mittel der Selbstdarstellung; er diente zur Prahlerei vor anderen, aber auch dem persönlichen Ausdruck, soweit dies innerhalb der Grammatik von Programmiersprachen möglich ist. LEVY beschreibt Programme als Ergebnisse kreativer Leistungen, die – wie alle solche Erzeugnisse – für Eingeweihte Rückschlüsse auf die Eigenarten und sogar den Humor ihrer Schöpfer zulassen:

> “Looking at Gosper’s programs, Sussman realized an important assumption of hackerism: all serious computer programs are expressions of an individual. [...] Sussman learned to read programs with the same sensitivity that a literature buff would read a poem. There are fun programs with jokes in them, there are exciting programs which do The Right Thing, and there are sad programs which make valiant tries but don’t quite fly.”[98]

GRAHAM zieht in seinem Buch „Hackers & Painters“ zahllose Parallelen zwischen Programmierern und bildenden Künstlern.[99] Beide seien kreativ, beide lernten ihre Tätigkeit vor allem durch Tun und weniger durch theoretisches Studium. Wie für Künstler sei auch für Programmierer die Orientierung an anderen Meistern des Fachs – und das Kopieren ihrer Werke und ihres Stils – eine wichtige Lehre. Das Arbeiten mit Skizzen bzw. Vorversionen sei ebenso typisch für Maler wie für Programmierer. Für beide sei es wichtig, die Perspektive ihres Publikums bzw. der Anwender im Auge zu behalten. GRAHAM kritisiert, daß die Tätigkeit des Programmierens dabei an den Hochschulen unter dem falschen Label „Computer science“ oder „Software engineering“ stehe – er sieht Programmierer weder als Wissenschaftler, noch als Ingenieure:

> “Along with composers, architects, and writers, what hackers and painters are trying to do is make good things. They’re not doing research per se, though if in the course of trying to make good things they discover some new technique, so much the better.”[100]

Mit der Regel „You can create art and beauty on a computer“ räumen sich Hacker die *Möglichkeit* ein, Kunst und Schönheit in Form von Computerprogrammen zu schaffen. Sie erklären diese Programme nicht grundsätzlich zur Kunst. Wie GRAHAM nämlich an anderer Stelle[101] ausführt, bezeichnet das Wort „hack“ nicht nur die brillante Lösung eines Problems, sondern auch ihr genaues Gegenteil: Den „häßlichen Hack“, der als unschöner, aber notwendiger Trick ein Programm ebenso zum Laufen bringt, wie ein Nylonstrumpf als Keilriemen-Ersatz die Lichtmaschine eines Autos.

---

97 Alle Zitate aus: LEVY, a.a.O., S.44f. Auch heute noch werden, vor allem im Unix-Lager, fähige Programmierer als „Gurus“ oder „Wizards“ bezeichnet.

98 Ebd., S.118.

99 Vgl. GRAHAM, a.a.O., S.18-33.

100 Ebd., S.18

101 Ebd., S.50.

1.3.6 Technischer Optimismus

LEVY sieht ein Grundverständnis als fundamental für alle von ihm beschriebenen Hacker-Gemeinschaften an:[102] „Computers can change your life for the better.“[103] In dieser Regel manifestiert sich der Glaube, daß Computer grundsätzlich etwas Gutes sind. Das war Anfang der 60er nicht selbstverständlich.[104]

Um den Optimismus zu erklären, der in dieser letzten Regel der Hacker-Ethik steckt, muß man sich vergegenwärtigen, daß die damaligen Hacker die ersten waren, die die universale Verwendbarkeit von Computern entdeckten *und sie daraufhin programmierten.* Bereits 1960 brachte Peter Samson den TX-0-Computer dazu, auf seinem Lautsprecher eine Melodie von Johann Sebastian Bach abzuspielen – das erste Mal in der Geschichte, daß Töne durch Zahlen und nicht direkt durch Vibrationen erzeugt wurden.[105] Bob Wagner schrieb ein Programm, das auf dem drei Millionen Dollar teuren TX-0 Funktionen von (weit billigeren) elektromechanischen Rechenmaschinen nachbildete – er nannte es entsprechend „Expensive Desk Calculator“.[106] Ein anderer implementierte einen „Expensive Typewriter“.[107] Steve Russell programmierte 1962 auf dem PDP-1-Computer „Spacewar“, das erste digitale Ballerspiel.[108] Alan Kotok und sein Professor John McCarthy stellten 1967 eine erste Schach-Software fertig;[109] ein weiteres Schachprogramm, Richard Greenblatts „MacHack“, besiegte 1969 in einer der ersten Mensch-Maschine-Schachpartien den AI-Forscher Herbert Dreyfus.[110]

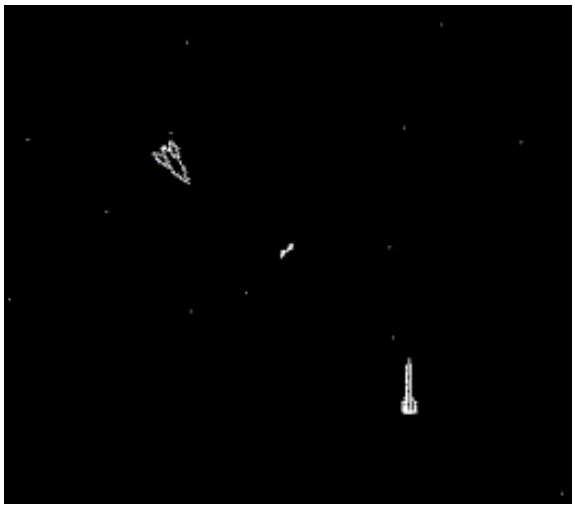


Abbildung 6 Spacewar – Videospiel von 1962[111]

---

[102] Vgl. etwa LEVY, a.a.O., S.67, 157, 175.

[103] Ebd., S.45.

[104] Zur gleichen Zeit entwarf der Regisseur Stanley Kubrick seine Vision von HAL 9000, dem kontrollsüchtigen Computer aus „2001: A Space Odyssey“.

[105] Vgl. LEVY, a.a.O., S.33f.

[106] Ebd., S.46f.

[107] Ebd., S.47.

[108] Ebd., S.65. Ein Klon des Programms kann im Web heute noch gespielt werden: http://lcs.www.media.mit.edu/groups/el/projects/spacewar/ [Aufgerufen am 05.09.2004].

[109] Vgl. LEVY, a.a.O., S.47f. Allerdings war der Computer bereits nach acht Zügen matt.

[110] Vgl. ebd., S.89f.

[111] Screenshot des oben genannten Klons.

Indem diese Studenten entdeckten, daß ein Computer kein Computer ist, sondern ein Klavier, eine Rechenmaschine, eine Schreibmaschine oder ein Schachspiel, nahmen sie eine Erfahrung vorweg, die die übrige Bevölkerung erst zwanzig bis dreißig Jahre später machte: „Everything around us is turning into computers."[112] Die – in den 60ern noch sehr viel sensationellere – Erfahrung dieser Allmacht führte bei den ersten Hackern zu einem tiefen Grundvertrauen zu „ihren" Maschinen und zu ihrem Fach:

> "They would be the ones who would guide computers to greater heights than anyone expected. And the hackers, by fruitful, meaningful association with the computer, would be foremost among the beneficiaries. But they would not be the only beneficiaries. Everyone could gain something by the use of thinking computers in an intellectual automated world."[113]

Geteilt wurde diese zutiefst positive Einstellung auch von ihren Professoren John McCARTHY und MINSKY. So meinte McCARTHY bereits 1962, es sei der natürliche Zustand des Menschen, permanent online zu sein: "What the user wants is a computer that he can have continuously at his beck and call for long periods of time."[114]

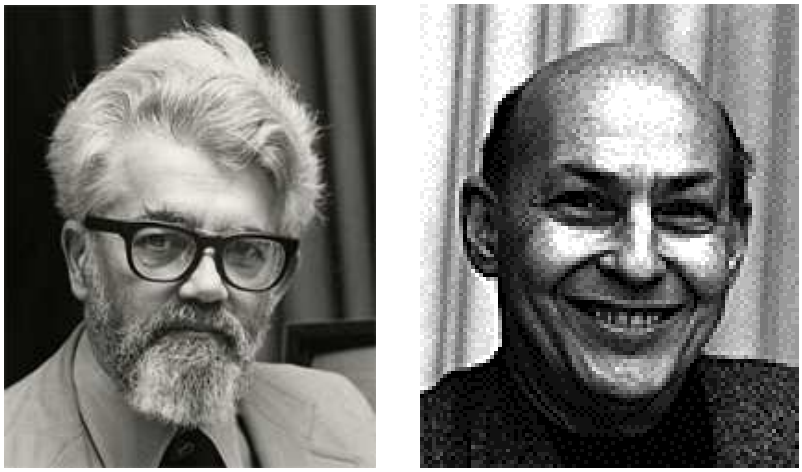


Abbildung 7: Die MIT-Professoren John McCarth und Marvin Minsky[115]

„Computers can change your life for the better" – zu den wenigen Gruppen, die Anfang der 60er diesem Glauben anhingen, gehörte neben Hackern, Professoren und einigen Science-fiction-Autoren auch das Militär. Im November 1962 gab es erste Gespräche mit der Forschungsagentur des US-Verteidigungsministeriums, der „Defense Department's

---

[112] GRAHAM, a.a.O., S.IX. Analog zu dieser Erkenntnis könnte man ergänzen, daß ein moderner Elektronikmarkt dem Wohnzimmer von König Midas ähnelt. Man sieht Lexika, Notizbücher, Telefone, Fotoapparate, Diavorträge und Straßenkarten und alles besteht aus dem gleichen Stoff: Aus Silikat und Software – aus Computern. Und man kauft die Straßenkarte, die am freundlichsten zu einem spricht.

[113] LEVY, a.a.O., S.49.

[114] McCARTHY, John: Time Sharing Computer Systems. Cambridge, Mass: MIT Press, 1962. Zitiert nach: LEVY, a.a.O., S.67.

[115] Das Bild von John McCarthy stammt von seiner Homepage an der Stanford-Universität unter: http://www-formal.stanford.edu/jmc/index.html [Aufgerufen am 10.01.2005].
Das Bild von Marvin Minsky ist auf seiner Universitätsseite beim MIT Media Lab veröffentlicht: http://web.media.mit.edu/~minsky/ [Aufgerufen am 10.01.2005].

Advanced Research Projects Agency“ (ARPA oder DARPA[116]) über eine Finanzierung der Computerforschung am MIT.[117] Drei Millionen Dollar jährlich auf langfristiger Basis sagte ARPA zu; auf dieser Zusage gründeten die Computerwissenschaftler am MIT, unter ihnen John McCARTHY und Marvin MINSKY, am 01. Juli 1963 das „Project MAC“, dessen Initialen sowohl „Machine Aided Cognition“ als auch „Multiple Access Computing“ bedeuteten.[118] Diese Doppeldeutigkeit machte auch durchaus Sinn: Ein Drittel des Geldes floß in MINSKYS Fachgebiet „Künstliche Intelligenz“, der Rest sollte für die Entwicklung von sogenannten Time-sharing-Systemen[119] verwendet werden. Nach Angaben des MIT erteilte ARPA weder konkrete Aufträge noch wurden die Wissenschaftler aufgefordert, spezielle militärische Anwendungen zu erstellen.[120] Auch ARPA selbst erklärte, die Gelder hätten der Grundlagenforschung gedient.[121] Allerdings räumte MINSKY gegenüber LEVY ein, daß seine Geldgeber durchaus militärisch verwendbare Ergebnisse sehen wollten und dies auch klar zum Ausdruck brachten:

> “There’s nothing illegal about a Defense Department funding research. It’s certainly better than a Commerce Department or Education Department funding research ... because that would lead to thought control. I would much rather have the military in charge of that ... the military people make no bones about what they want, so we’re not under any subtle pressures. It’s clear what’s going on. The case of ARPA was unique, because they felt that what this country needed was people good in defense technology. In case we ever needed it, we’d have it.’”[122]

Von 1963 bis 1973 unterstützte das US-Verteidigungsministerium über ARPA großzügig die Computerforschung am MIT und ermöglichte es dem Project MAC, jeden als Assistenten einzustellen, der Talent zum „Hacking“ zeigte.[123] Wegen dieser engen Verbindung wurde im Rahmen der amerikanischen Vietnamkriegs-Proteste 1969 auch gegen das Computerlabor am MIT demonstriert.[124] Das mangelnde Verständnis der Hacker für diese

---

[116] ARPA war der erste Name der Agentur, die 1958 nach dem Sputnik-Schock entstand. 1972 wurde das D für „Defense“ in den Namen hinzugenommen, 1993 nannte man sich wieder ARPA und seit 1996 wieder DARPA. Ihre Aufgabe ist es, „Forschung und Entwicklung in Bereichen voranzutreiben, in denen sowohl das Risiko als auch die Kosten sehr hoch sind, und in denen ein Erfolg zu dramatischen Vorteilen für das Militär in seinen traditionellen Rollen und Aufgaben führen kann“. Zitiert (und übersetzt) aus der Selbstdarstellung der DARPA-Homepage: http://www.darpa.mil/ [Aufgerufen am 01.11.2004]
Vgl. auch: WIKIPEDIA (engl.): Darpa. In: http://en.wikipedia.org/wiki/Darpa [Aufgerufen am 01.11.2004]

[117] „‘My decision to start Project MAC was made on Thanksgiving Day 1962, after extensive talks with J.C.R. Licklider, who was then at DARPA,’ Fano said in a recent interview in his old Tech Square office filled with orange moving crates. ‘The next day, Friday, I met with Provost [Charles] Townes, and told him what we had in mind. He said »Go ahead«’“ Aus: MIT Tech Talk: MIT leaves behind a rich history in Tech Square, 17.03.2004 (Volume 48, Number 21). Zitiert nach der Online-Ausgabe unter: http://web.mit.edu/newsoffice/2004/techsquare-0317.html [Aufgerufen am 02.10.2004].

[118] Vgl. LEVY, a.a.O., S.68. Der erste Direktor, Robert Fano, war über seine Arbeit am MIT Militärlabor Lincoln Lab mit ARPA verbunden. Das Project MAC teilte sich 1970 in das AI Lab (das zwar seit 1959 existiert hatte, aber keine eigenständige Einheit war) und das Laboratory for Computer Science.

[119] Mehrbenutzerfähige Programme wie z.B. Unix.

[120] LEVY, a.a.O., S.131.

[121] LEVY zitiert mit dieser Aussage den früheren Vorsitzenden der ARPA-Büros für Datenverarbeitungstechnik (IPTO), Robert Taylor. Vgl. ebd., S.131.

[122] Ebd., S.131. LEVY zitiert hier wörtlich MINSKYs Aussagen. Die Auslassungen (...) entsprechen dem Original und sollen offenbar Denkpausen symbolisieren.

[123] Vgl. ebd., S.150f.

[124] Vgl. ebd., S.132f.

Demonstrationen sieht LEVY als Anzeichen dafür, wie weit sie sich mit ihrem technischen Optimismus vom Durchschnitt ihrer Gesellschaft entfremdet hatten.[125]

Der Glaube, daß Computer das Leben verbessern, war auch ein wesentliches Motiv für die spätere Bildung der Community der „Hardware-Hacker“ an der amerikanischen Westküste; hier allerdings war dieser Glaube verbunden mit linken und eher revolutionären Parolen: „Computer Power to the People!“[126] war einer der Slogans. Auch der „Homebrew Computer Club“, zu dem Experten wie Apple-Gründer Steve Wozniak gehörten, war ursprünglich von seinem Gründer Fred Moore als soziale Bewegung gedacht gewesen.[127]

In der sechsten Regel der Hacker-Ethik zeigt sich also ein grundsätzlicher, gelegentlich blauäugiger Optimismus bezüglich der Bedeutung des eigenen Fachgebiets; die Weiterentwicklung der Computer- und Informationstechnik wird dabei verbunden mit einer sozialen Utopie. Möglicherweise erklärt das auch, warum sich Computerexperten in stärkerem Maße als Autofans sozial engagieren: In den 70ern etwa in Zusammenhang mit dem Öko-Magazin „Whole Earth Catalog“, in den 80ern für die „Electronic Frontier Foundation“; in Deutschland für das „Forum InformatikerInnen für Frieden und gesellschaftliche Verantwortung“ (FIfF)[128] und heute etwa in den Organisationen „Foundation for a Free Information Infrastructure (FFII)“ und der „Free-Software-Foundation“ (FSF).

Abbildung 8: Cover (links) von Ted Nelsons Broschüre „Computer Lib“ aus dem Jahr 1974.[129] „You can and must understand computers NOW!“ steht als Motto über dem Titel: Nelson sah Computer als Instrumente der Aufklärung. Als Mittel der Beschwichtigung sah sie dagegen FIfF: Das „Forum InformatikerInnen für Frieden und gesellschaftliche Verantwortung“ warnte 1985 davor, sich während der kurzen Vorwarnzeiten vor einem Atomschlag auf die Entscheidungskompetenz von Großrechnern zu verlassen (rechts).[130] Gemeinsam war ihnen das Bewusstsein, durch die Beherrschung des Computers zu einer gesellschaftlichen Elite mit besonderer Verantwortung zu gehören.

---

125 Vgl. ebd., S.133.

126 NELSON, Theodor H.: Computer Lib/Dream Machines: New Freedom Through Computer Screens – a Minority Report. Chicago: Hugo's Book Service: 1974. Zitiert nach: ebd., S.175.

127 Vgl ebd., S.213f.

128 Vgl. BORCHERS, Detlef: Verteiltes Nachdenken. 20 Jahre kritische Informatik. In: c't Magazin für Computertechnik. Ausgabe 22/2004. Hannover: heise, 18.10.2004, S. 110-111.

129 NELSON, a.a.O., Titelseite.

130 Plakat aus der Ausstellung „20 Jahre FIfF“ vom 12. bis zum 28.02.2005 in Bremen. Foto: Ulrich Hansen.

1.4 Zusammenfassung

Mit der Hacker-Ethik entstand das Selbstverständnis eines neuen Wissenschafts- und Berufsfeldes. Aus der Interferenz der Gruppe der Modelleisenbahn-Bastler, also der „Technikfreaks“, mit anderen Bereichen wie der theoretischen Mathematik und den Ingenieurwissenschaften bildete sie sich als neue Wissenskultur heraus, wie sie aus techniksoziologischer Sicht zur Produktion neuen technischen Wissens erforderlich ist.[131] Die Hacker-Kultur entstand dabei nicht zufällig, sondern als Ergebnis eines bewußten Experiments der Verantwortlichen am AI Lab, eine neue technische (und keineswegs künstliche) Intelligenz zu fördern; ein „Paradies“[132] wissenschaftlicher Freiheit, das auf den gewaltigen Finanz- und Sachmitteln basierte, die das US-Verteidigungsministerium diesem Think-Tank vor dem Hintergrund des Kalten Kriegs gewährte.

In dieser Atmosphäre wurden Entwicklungsmöglichkeiten angedacht, die heute noch durch ihre Modernität beeindrucken – etwa die Beschäftigung mit neuronalen und semantischen Netzen oder der hochabstrakten Programmiersprache Lisp[133] – oder die aufgrund von Skurrilität und Kühnheit faszinieren, wie die Kopplung einer TV-Kamera mit einem PDP-6-Computer und einem mechanischen Arm sowie die anschließenden endlosen Versuche, mit diesem Roboter Pingpong zu spielen.[134]

Welche sozio-technischen Konstrukte die neue Wissenskultur hervorbrachte, zeigt das ***„Incompatible Time-sharing System“*** (ITS), ein Betriebssystem, das LEVY als die Verkörperung der Hacker-Ethik bezeichnet: Es gab keine Paßwörter, jeder Benutzer hatte Zugang zu allen Daten jedes anderen Benutzers und um das System zum Absturz zu bringen genügte der Befehl „Kill“.[135] Eben weil diese Möglichkeiten vorhanden waren, wurden sie von den Benutzern nicht mutwillig angewandt: „The idea was to take all fun away from crashing the system by making it trivial to do so.“[136] Das System appellierte erfolgreich an die soziale Kompetenz aller Beteiligten und ist darin möglicherweise vergleichbar mit dem viel später entstandenen Wikipedia-Projekt.[137]

---

131 Vgl. DIERKES, Meinolf; HOFFMANN, Ute; MARZ, Lutz: Leitbild und Technik. Zur Entstehung und Steuerung technischer Innovationen. Berlin: Edition Sigma, 1992, S. 29-41.

132 Vgl. GREVE, Georg C. F.: Geschichte und Philosophie des GNU Projektes. Vortrag an der Universität Paderborn, 05. 12.1998. Online veröffentlicht unter: http://www.gnu.org/philosophy/greve-clown.de.html [Aufgerufen am 09.12.2204].

133 GRAHAM zufolge ist diese Sprache selbst im Jahr 2004 noch die „Dream Language“ eines jeden Hackers. Vgl. GRAHAM, a.a.O., S. 200-215.

134 Vgl. MINSKY, Marvin: Television Camera-To-Computer Adapter: PDP-6 Device 770. Artificial Intelligence Project Memo No 75 (AIM-75), January 1965. Online veröffentlicht unter: ftp://publications.ai.mit.edu/ai-publications/pdf/AIM-075.pdf [Aufgerufen am 02.10.2004].

135 Vgl. LEVY, a.a.O., S.123-128.

136 Ebd., S.125.

137 Ein Versuch, für das System weitere Interessenten zu finden, scheiterte 1968 aufgrund der beschriebenen Eigenschaften. Am AI Lab jedoch setzte sich ITS durch und wurde immerhin bis 1982 verwendet und verbessert. Seine Existenz endete dann, da Digital Equipment (DEC) zu diesem Zeitpunkt die Entwicklungslinie seiner PDP-Computer einstellte: ITS war in Assemblersprache für PDP-Computer programmiert, hing also von dieser Hardware ab. (Zu Assemblersprachen siehe Anhang: „Das Schwein guckt ins Uhrwerk oder: Was ist Quellcode?“)

2 Die Free-Software-Bewegung

Wie in den meisten Fundstellen, in denen das Wort „Free Software“ in seiner eigentlichen Bedeutung verwendet wird, ist es auch an dieser Stelle zunächst nötig, darauf hinzuweisen, daß damit nicht „Verschenksoftware“ oder sogenannte „Freeware“ gemeint ist. Statt dessen handelt es sich bei dem Begriff um eine Neudefinition von Software als geistiger Schöpfung und als informationeller Infrastruktur von Hackern. Dies war nötig geworden, weil sich das Verständnis des Wortes „Software“ im Laufe der Zeit stark verändert hatte: Der Begriff wurde erstmals 1957 verwendet,[138] damals wurde durch den Wortbestandteil „Ware“ noch keine kommerzielle Bedeutung ausgedrückt.[139] Das sollte sich Mitte der 70er Jahre ändern.

Im folgenden werden die Veränderungen im Verständnis von Software beschrieben, die sich vor allem durch die massenhafte Verbreitung der Hardware ergaben. Am Beispiel der Hacker-Community am MIT und ihrem Protagonisten Richard Stallman werden dann die Konsequenzen gezeigt, die diese Veränderungen für das Selbstverständnis der Programmierer, d. h. für die Hacker-Ethik hatten. In einem dritten Punkt geht es um die von Stallman initiierte Gegenbewegung, die „Free-Software“-Bewegung und ihre Methoden, die Hacker-Ethik gegen diesen Trend zu retten.

2.1 Die Kommerzialisierung von Software

Mit dem 1973 vom amerikanischen Kongreß verabschiedeten Mansfield-Amendment wurde es für das US-Verteidigungsministerium und damit auch für ARPA schwieriger, Ausgaben zu tätigen, die nicht in direktem Zusammenhang mit Verteidigungsprojekten standen[140] (wiewohl die Zusammenarbeit mit dem MIT damit keineswegs endete[141]). Wie andere Universitäten und Institute begann deswegen auch das AI Lab zunehmend mit privaten Investoren zusammenzuarbeiten. Diese Kooperationen und der mit der wirtschaftlichen Bedeutung von Computern wachsende Bedarf an Programmierern führten dazu, daß im Jahr 1980 die meisten Mitarbeiter am AI Lab, unter ihnen viele Hacker, in Teilzeit an

---

[138] John Wilder Tukey verwendete 1957 als Erster den Begriff „Software“ als Abgrenzung zu „Hardware“. Letzterer Begriff entspricht dem früher gebräuchlicheren deutschen Wort „Eisenwaren“ oder dem österreichischen „Hartwaren“ und wird im englischen Sprachraum nicht nur im Zusammenhang mit Computern verwendet. Vgl.: WIKIPEDIA (engl.): Software. In: http://en.wikipedia.org/wiki/Software [Aufgerufen am: 03.10.2004].

[139] Wie bereits im Kapitel zur Entstehung der Hacker-Kultur beschrieben, wurde Software in den frühen 60ern in seinem Entstehungszusammenhang, etwa an Universitäten, in großen Firmen oder Militäreinrichtungen nicht als Ware betrachtet, sondern als Teil des Computersystems und war als solches für jeden zugänglich, der Zugang zum Computer hatte. Wie BAUMGÄRTEL schreibt, hätten die Computerhersteller damals Software verstanden als „eine Art Service oder ‘Dreingabe’ zu den Computern, die das Unternehmen verkaufte“. BAUMGÄRTEL, Tilman: Am Anfang war alle Software frei. Microsoft, Linux und die Rache der Hacker. In: ROESLER, Alexander; STIEGLER, Bernd (Hrsg.): Microsoft. Medien, Macht. Monopol. Frankfurt a. M.: Suhrkamp, 2002, S. 103-129, hier S. 110.

[140] Vgl. LEVY, a.a.O., S.150. Siehe auch: HAUBEN, Ronda: Über die Entstehung des Internet und die Rolle der Regierung. In: Telepolis – Magazin der Netzkultur, 24.10.1999. http://www.heise.de/tp/deutsch/inhalt/co/5409/1.html [Aufgerufen am 03.10.2004].

[141] So scheint das von ARPA Mitte der 70er am MIT angestoßene Projekt der Spracherkennung noch immer aktiv zu sein. Vgl. c’t Magazin für Computertechnik: „Unbemannte Flugzeuge per Sprache steuern“. Ausgabe 24/2004. Hannover: heise, 15.11.2004, S. 54.

kommerziellen Projekten arbeiteten – mit negativen Auswirkungen auf die Hacker-Gemeinschaft:

> "What at first seemed like a win-win deal – hackers got to work on the best projects, giving the lab first look at many of the newest computer technologies coming down the pike – soon revealed itself as a Faustian bargain. The more time hackers devoted to cutting-edge commercial projects, the less time they had to devote to general maintenance on the lab's baroque software infrastructure. Soon, companies began hiring away hackers outright in an attempt to monopolize their time and attention. With fewer hackers to mind the shop, programs and machines took longer to fix."[142]

Programmieren hatte sich von einer forschenden Tätigkeit zu einer gefragten Dienstleistung entwickelt. Eine ähnliche Veränderung gab es beim Produkt dieser Tätigkeit, der Software.

BAUMGÄRTEL macht die Firma Microsoft für den „Paradigmenwechsel"[143] verantwortlich, der in den 70er Jahren im Verständnis von Software und der Arbeit von Programmierern stattgefunden hat. „Einer Kultur der Schließung bei Microsoft steht einer Kultur der Offenheit in der Hackerszene gegenüber, und die Ansicht, daß Computer und ihre Programme Kultur sind, die jedem zugänglich sein soll."[144] Diese Veränderung habe mit der Gründung der Firma begonnen, deren Geschäftsidee BAUMGÄRTEL zufolge darin bestand, „daß Gates Software zu einem Produkt gemacht hatte, das zu hohen Preisen verkauft wurde"[145].

Tatsächlich gibt es im Zusammenhang mit Microsoft ein Ereignis, das als erster bekannter „Clash of Culture" zwischen Kommerz- und Hacker-Kultur angeführt werden kann: Die Raubkopie des von Bill Gates und Paul Allen geschriebenen BASIC-Interpreters für den Altair 8800 durch den Homebrew Computer Club im Jahr 1975.

Der Altair 8800 war mit einem Preis von 397 Dollar der erste Computer, der für Durchschnittsverdiener erschwinglich war und gilt als Wegbereiter des PC.[146] Im Januar 1975 wurde dieser „Mikrocomputer" im Magazin „Popular Electronics" angekündigt und fand enorme Resonanz.[147] Wenige Tage später nahmen der 19jährige Gates und der zwei Jahre ältere Allen Kontakt mit der Herstellerfirma Micro Instrumentation Telemetry Sys-

---

142 WILLIAMS, a.a.O., S.92.

143 BAUMGÄRTEL, a.a.O., S. 108.

144 Ebd.

145 Ebd. Es ist hier anzumerken, daß das Betriebssystem DOS nicht zu hohen, sondern zu niedrigen Preisen verkauft wurde. PC-DOS 1.0 wurde von IBM für 40$ an Endkunden abgegeben, wesentlich billiger als das konkurrierende CP/M (240$). Die Lizenzkosten berechnete Microsoft pro Installation, pro System oder (als billigste Variante) pro Prozessor, wodurch andere große PC-Hersteller die Möglichkeit hatten, ihre Kosten auf einen Bruchteil des Einzelpreises zu senken. Für alle Rechte an DOS zahlte Microsoft dem Programmierer Tim Paterson zunächst 25.000 US-Dollar, später weitere 50.000 US-Dollar.
Vgl. WIKIPEDIA (engl.): QDOS. In: http://en.wikipedia.org/wiki/QDOS sowie WIKIPEDIA (engl.): Bill Gates. In: http://en.wikipedia.org/wiki/Bill_Gates [Beide aufgerufen am 04.11.2004].

146 Das Gerät basierte auf dem Intel 8080-Chip, hatte 256 Byte RAM, für Eingaben mußte man Hebel an der Frontseite drücken, und statt einem Monitor hatte der Rechner rote Lämpchen. Er wurde außerdem als „Kit" geliefert; das heißt der Benutzer mußte ihn selbst zusammenbauen. Vgl. LEVY, a.a.O., S.187-200 sowie unter: WIKIPEDIA (engl.): Altair 8800. http://en.wikipedia.org/wiki/Altair_8800 [Aufgerufen am 06.11.04.

147 Der Hersteller hatte die Produktion von 400 Exemplaren geplant. Bereits im ersten Monat wurden mehrere tausend Geräte verkauft. Vgl. LEVY, a.a.O., S.190ff.

tems (MITS) auf. Ihr boten sie einen BASIC-Interpreter[148] an, wobei sie verschwiegen, daß zu diesem Zeitpunkt keine Zeile geschrieben war.[149] MITS willigte ein und nur acht Wochen später präsentierte Allen eine Vorversion, die er und Gates unter einem selbstgebastelten Altair-Simulator auf dem leistungsfähigeren PDP-10-Computer ihrer Hochschule programmiert hatten.[150] Obwohl es bis zum Herbst dauern sollte, bis Altair-BASIC offiziell verfügbar war, nahm MITS bereits im Frühjahr Bestellungen an.

Abbildung 9: Der Altair 8800 auf dem Titelblatt von „Popular Electronics", Januar 1975.

Die Zeitspanne bis zur Auslieferung von Gates' Altair-Basic schien vielen Computerfans zu lang, vor allem einigen Mitgliedern des „Homebrew Computer Club", einer Gemeinschaft von Bastlern und Computerfans in Silicon Valley, die LEVY als „Hardware-Hacker"[151] bezeichnet. Bei den wöchentlichen Sitzungen wurden Elektronikteile, Diagramme und Software ausgetauscht, vor allem aber Ideen – so erklärt etwa Apple-

---

[148] Die Computersprache „Beginners All-Purpose Symbolic Instruction Code" (BASIC) war von den beiden Hochschuldozenten John George Kemeny und Thomas Eugene Kurtz am Dartmouth-College in einer fast zehnjährigen Entwicklungstätigkeit erdacht und 1964 frei veröffentlicht worden; es war die Sprache, die Gates und Allen selbst im Computerunterricht erlernt hatten. Ein Interpreter setzt die Befehle der Programmiersprache in die Instruktionen des jeweiligen Prozessors um, für den er geschrieben wurde und führt sie aus.

[149] Beide konnten aber zu diesem Zeitpunkt beträchtliche Erfahrung mit Computern vorweisen. Bereits in der Highschool hatten sie gemeinsam für Computerfirmen Software geschrieben und administriert und sogar eine kleine Firma gegründet, Traf-o-Data, die ein Gerät zur Messung des Straßenverkehrsflusses herstellte. Vgl. MIRRICK, John: Before Microsoft. 29.09.1996. Webseite der Fakultät für Computerwissenschaft der Polytechnischen Hochschule Virginia:
http://ei.cs.vt.edu/~history/Gates.Mirick.html [Aufgerufen am 06.11.2004].
MIRRICK verweist zu diesem Thema vor allem auf: WALLACE, James: Hard Drive: Bill Gates and the Making of the Microsoft Empire, New York: HarperCollins Publishers, 1993.

[150] Vgl. HAGEN, Wolfgang: Bill Luhan und Marshall McGates. Die Extension des Menschen als Extension der USA. In: ROESLER; STIEGLER (Hrsg.), a.a.O., S. 11-47, hier S. 33.

[151] Vgl. LEVY, a.a.O., S. 153-278.

Erfinder WOZNIAK: „Without computer clubs there would probably be no Apple computers. Our club in the Silicon Valley, the Homebrew Computer Club, was among the first of its kind."[152]

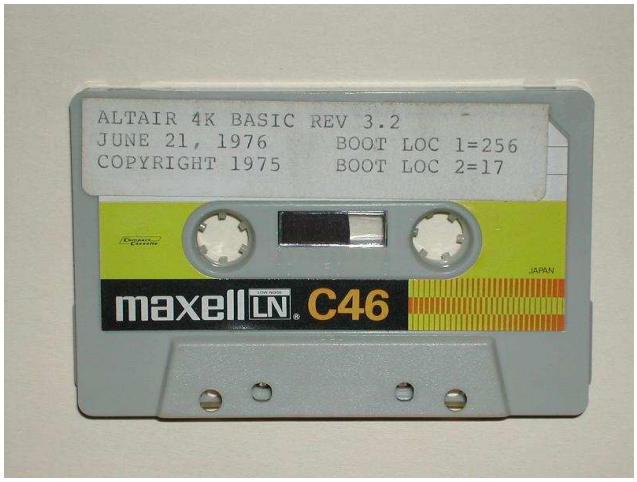


Abbildung 10: Original-Software von Microsoft – Altair-Basic zum Kaufpreis von 150 Dollar[153]

Als im Juni 1975 immer noch kein Altair-BASIC ausgeliefert wurde, nahmen zwei der Hardware-Hacker von Homebrew die Sache selbst in die Hand: John Draper, der auch als „Captain Crunch"[154] Berühmtheit erlangte, stahl bei einer Präsentations-Show von MITS eine Kassette mit einer Vorversion von Altair-BASIC. Zum nächsten Clubtreffen brachten er und Dan Sokol einen Karton Kopien mit und verteilten sie kostenlos – unter der Bedingung, daß jeder, der dieses Mal eine Kassette erhielt, beim nächsten Treffen zwei weitere mitbringen müsse.[155] Das Programm verbreitete sich weit über Silicon Valley hinaus.

---

[152] WOZNIAK, Stephen: Homebrew and how the Apple came to be. In: DITLEA, Steve (Hrsg.): Digital Deli – The comprehensive user-lovable menu of computer lore, culture, lifestyles and fancy. New York: Workman Publishing, 1984. Zitiert nach der Online-Ausgabe auf: http://www.atariarchives.org/deli/ [Aufgerufen am 06.11.2004].
In einem späteren Interview sagte er über Homebrew: „Ich kam aus einer Gruppe, die man als Beatniks oder Hippies bezeichnen könnte – ein Haufen Techniker, die radikale Reden über eine Informationsrevolution schwangen und darüber, wie wir die Welt völlig verändern und Computer in jeden Haushalt bringen würden." KENNEDY, John: Steve Wozniak: Hacker and Humanitarian. In: KAWASAKI, Guy (Hrsg.): Hindsights – The Wisdom and Breakthroughs of Remarkable People. 1994. Zitiert nach: HIMANEN, Pekka: Die Hacker-Ethik und der Geist des Informationszeitalters. München: Riemann, 2001, S. 215.

[153] Quelle: http://www.widomaker.com/~cswiger/altair.html [Aufgerufen am 31.01.2005].

[154] Er kam zu diesem Namen, weil er entdeckt hatte, daß eine Pfeife, die der Frühstücksflockenpackung gleichen Namens beilag, einen Ton von der Höhe erzeugte, die im Telefonsystem zur Herstellung von Auslandsverbindungen genutzt wurde. Draper war damals bei der US-Luftwaffe im Ausland stationiert und nutzte seine Entdeckung, um kostenlos nach Hause zu telefonieren. Vgl. LEVY, a.a.O., S. S.248.

[155] Vgl. ebd., S.228f.

Abbildung 11: Bill Gates im Alter von 21 Jahren bei der Eröffnungsansprache zur Computermesse „Altair World Fair" 1976.[156]

Nachdem Gates zunächst angenommen hatte, die Nutzer der raubkopierten Vorversion würden möglicherweise doch noch eine fehlerbereinigte offizielle Version kaufen, reagierte er auf den Diebstahl[157] am 3. Februar 1976 mit seinem berühmt gewordenen „Open Letter to Hobbyists":

> "[...] Almost a year ago, Paul Allen and myself, expecting the hobby market to expand, hired Monte Davidoff and developed Altair BASIC. [...] The value of the computer time we have used exceeds $40,000.[158] The feedback we have gotten from the hundreds of people who say they are using BASIC has all been positive. Two surprising things are apparent, however, 1) Most of these "users" never bought BASIC (less than 10% of all Altair owners have bought BASIC), and 2) The amount of royalties we have received from sales to hobbyists makes the time spent on Altair BASIC worth less than $2 an hour. Why is this? 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? Is this fair? [...] One thing you do do is prevent good software from being written. Who can afford to do professional work for nothing? What hobbyist can put 3-man years into programming, finding all bugs, documenting his product and distribute for free? [...] Most directly, the thing you do is theft."[159]

---

[156] Die Messe in Albuquerque, New Mexico, drehte sich ausschließlich um den Altair 8800-Computer. Quelle des Fotos: FREIBERGER, Paul; SWAINE, Michael: Fire in the Valley. The Making of the Personal Computer. New York: McGraw-Hill, 1984/2000. Online veröffentlicht unter: http://www.fireinthevalley.com/fitv_voices.html [Aufgerufen am 10.10.2005].

[157] Darum handelte es sich zweifellos, nach dem damals wie heute geltenden Urheberrecht, auch wenn die Computerfans damals sehr empfindlich auf den Begriff reagierten. LEVY beschreibt, daß Mitglieder von Homebrew den Vorfall damit rechtfertigten, daß viele die Software ohnehin bereits bestellt und per Vorkasse bezahlt hatten, aber von MITS im Dunkeln über den Lieferzeitpunkt gelassen wurden. Vgl. ebd., S.228.

[158] Diese Behauptung ist aus heutiger Sicht eher mit Humor zu betrachten, da der Großteil von Altair Basic auf dem PDP-10-Computer der Harvard-Universität entstanden ist. Gates wurde später deswegen aus Harvard ausgeschlossen. So schreibt sein damaliger Dozent Christos PAPADIMITRIOU: „He was expelled for 'stealing' computer time in Harvard's PDP20, then a valuable resource." Zitiert nach einer Email, veröffentlicht in: HAGEN, a.a.O., S. 30. PAPADIMITRIOU bezeichnet die PDP-10 mit ihrem Spitznamen PDP-20, den sie erhalten hatte, weil sie unter dem Betriebssystem DECSYSTEM-20 lief. Eine PDP-20 wurde nie gebaut.

[159] GATES III, William Henry: An Open Letter to Hobbyists. Altair Users' Newsletter. Albuquerque: 1976. Zitiert nach: http://www.blinkenlights.com/classiccmp/gateswhine.html [Aufgerufen am 09.11.2004].

Gates erhielt als Reaktion rund 400 Zuschriften, die meisten von ihnen in empörtem Tonfall; die Verkäufe steigern konnte sein offener Brief aber nicht.[160] Bedeutung erlangte dieser in anderer Hinsicht: LEVY sieht das Ereignis als „software flap“[161] – als Wendepunkt im Verständnis von Software.

Es wäre allerdings blauäugig, allein Bill Gates und seine Geschäftstüchtigkeit zur Ursache dieser Wende zu stilisieren. Statt dessen ist festzustellen, daß eine kommerzielle Bedeutung von Software überhaupt erst mit der massenhaften Verbreitung der Hardware entstehen konnte: Vom DEC PDP-1-Computer, an dem die MIT-Hacker programmierten, wurden nur fünfzig Stück hergestellt, den PDP-6 gab es nur zwanzigmal; kein Wunder also, daß in den elitären Zirkeln der Nutzer dieser Rechner die Software frei zirkulieren durfte. Vom Altair 8800 wurden dagegen 4.000 Exemplare[162] gebaut und verkauft. Mit diesem Minicomputer entstand zum ersten Mal in der Computergeschichte ein Markt für Software und es gab nicht mehr nur – wie zuvor – einen auf Großfirmen und das Militär als Kunden beschränkten Markt für Programmierdienstleistungen. Und mit dem individuellen Besitz von PCs – statt geteilten Großrechnern – wurde auch Software zum individuellen Besitz. Bill Gates war nur der erste, der diesen Markt erkannte:

> „Wir wußten, daß Mikroprozessoren mächtiger und zugleich billiger werden würden, so daß unweigerlich die Kosten für Computer heruntergehen mußten. Das aber, so fanden wir heraus, würde sie für kleine Unternehmungen, Studenten und für den normalen Haushalt erschwinglich machen. Und wir schlossen weiter, daß dies wiederum einen ungeheuren Bedarf an Software erzeugen würde.“[163]

Was Microsoft da vorhersah (und was heute wenig überraschend scheint) blieb anderen Computerpionieren zur gleichen Zeit verborgen: Steve Jobs und Steve Wozniak schrieben – möglicherweise beeinflußt von der Hacker-Ethik im Homebrew Club – noch im April 1976 in ihrer Werbung für den Apple I: „Our philosophy is to provide software for our machines free or at minimal cost.“[164]

---

[160] Trotz ihrer Niederlage im Kampf mit den Raubkopierern hatte die Firma Micro-Soft (so die damals korrekte Schreibweise) indirekt von dem Vorfall profitiert: „‘[...] the fact, that everybody had Altair BASIC and knew how it worked and how to fix it meant that when other computer companies came on line and needed a BASIC, they went to Gates’ company. It became a de facto standard.’“ LEVY, a.a.O., S.230. LEVY zitiert hier eines der ersten Mitglieder des Homebrew Computer Clubs, Steve DOMPIER, aus einem selbst geführten Interview.
Möglicherweise war das Altair-Geschäft für Gates und Allen auch eine wichtige Lehre in Sachen Lizenzrecht: Während sie Altair BASIC noch exklusiv an MITS lizenziert hatten und eine fixe Summe pro verkauftem Exemplar erhalten sollten, verliehen sie vier Jahre später dem ungleich größeren Geschäftspartner IBM nur eine nicht-exklusive Lizenz an MS-DOS und begründeten damit ihren Erfolg.

[161] LEVY, a.a.O.,.230.

[162] Diese Exemplare wurden alle 1975 gefertigt – ein Jahr später verschwand das Produkt dann vom Markt: „By 1976 there were a number of much better built machines on the market, and when Roberts started demanding the newly-appearing computers stores sell only Altair machines, they instead turned to the competition and in a turn of irony MITS was quickly squeezed out of the market they themselves had created.“ WIKIPEDIA (engl.): Altair 8800. In: http://en.wikipedia.org/wiki/Altair_8800, [Aufgerufen am 10.11.2004]. Die Zahl 4.000 ist möglicherweise eine Schätzung, ich berufe mich dabei auf:
https://www.ssl-id.de/www.reuter2000.de/u_itgeschichte.php [Aufgerufen am 10.11.2004].

[163] MICROSOFT Corporation: Inside out. Microsoft – In Our Own Words. New York: Warner Books, 2000, S. 7. Zitiert nach: HAGEN, Wolfgang: Bill Luhan und Marshall McGates. Die Extension des Menschen als Extension der USA. In: ROESLER, Alexander; STIEGLER, Bernd: Microsoft. Medien, Macht. Monopol. Frankfurt a. M.: Suhrkamp, 2002, S. 33.

[164] LEVY, a.a.O., S. 253.

Abbildung 12: Steve Jobs und Steve Wozniak 1975 mit einer „blue box“, einem Gerät zur Manipulation von Telefonverbindungen.[165]

Ironischerweise war es dann der Apple II, für den zum ersten Mal in großem Maßstab kommerzielle Software verkauft wurde. VisiCalc, die erste Tabellenkalkulation der Softwaregeschichte, wurde 1978 von Dan Bricklin und Bob Frankston entwickelt und allein 1979 900.000mal verkauft – zum Einzelpreis von 100 Dollar.[166] Gemeinsam mit der ebenfalls 1978 entstandenen Textverarbeitung WordStar sorgte VisiCalc für den Einzug von Minicomputern in die Büros der Unternehmen und etablierte die Klasse der Standardprogramme – und damit das heutige Verständnis von Software.

Der Erfolg von VisiCalc stellte für IBM den Anstoß dar, den PC zu entwickeln. Das gab der Firma Micro-Soft, die Gates und Allen am 26. November 1976 zur Vermarktung von Altair BASIC gegründet hatten, ein zweites Mal die Chance, die die Firma bereits 1976 zu ergreifen gehofft hatte.

Interessant an dieser Episode der Computergeschichte ist, daß die Vorgehensweise von Gates und Allen bei der Entwicklung und Vermarktung von Altair BASIC die spätere Geschäftspraxis von Microsoft in groben Zügen vorwegnahm: Die Verwertung, der Kauf oder die Kopie bereits etablierter Software,[167] die frühe Ankündigung und Vermarktung von Vaporware – also nichtexistenten Produkten – um Wettbewerber auszuschließen oder zu entmutigen,[168] und möglicherweise sogar das Setzen von Standards durch das Dulden privater Raubkopien und das Verschenken von Software.[169]

All diesen Strategien ist jedoch gemeinsam, daß sie wenig nutzen gegenüber freier Software. Denn die im „Open Letter to Hobbyists“ rhetorisch offen gelassene Frage: „Who can afford to do professional work for nothing?“ wurde von dieser Idee auf unerwartete Weise beantwortet.

---

[165] Quelle: Homepage von Steve Wozniak, http://www.woz.org/ [Aufgerufen am 10.10.2005].

[166] Vgl. The History of Computing Project: http://www.thocp.net/timeline/1978.htm sowie Dan Bricklins Website: http://www.danbricklin.com [Beide aufgerufen am 10.11.2004].

[167] Außer im Fall von BASIC etwa geschehen bei DOS (Klon von CP/M), Windows (Xerox/PARC), Word (Xerox Bravo), Excel (VisiCalc), Internet Explorer (Mosaic).

[168] Ähnlich geschehen etwa bei der frühen Ankündigung von Windows 95, mit negativen Effekten möglicherweise für das konkurrierende OS/2.

[169] Verschenkt wurden etwa der Internet Explorer, Outlook Express und der Media Player.

2.2 Stallman und der Bruch mit der Hacker-Ethik am MIT

Die Entstehung von Nachfragemärkten für Software und Programmierdienstleistungen begann Anfang der 80er auch das „Hacker-Utopia“ am MIT zu beeinträchtigen. Im Zuge dieses Wandels gelang es den Hackern am AI Lab nicht mehr, ihre Hacker-Ethik aufrechtzuerhalten. Betroffen von diesen Entwicklungen waren vor allem die ersten drei Regeln der Hacker-Ethik:

- Die Zahl der Nutzer von Softwareprogrammen hatte gegenüber den 60er Jahren erheblich zugenommen. Die wenigsten dieser Nutzer waren „Hacker“. Sie betrachteten es als selbstverständlich, daß Sicherheitsmaßnahmen wie Paßwörter existierten und daß persönliche Daten vor den Augen anderer geschützt wurden. Mit der Einrichtung entsprechender Systeme am AI Lab wurde die dritte Regel der Hacker-Ethik „Mistrust Authority – promote decentralization“ gebrochen sowie der zweite Grundsatz „All information should be free“.
- In das AI Lab drang zunehmend Software ein, die proprietär, also nicht im Quellcode verfügbar war. Diese Software brach die erste Regel der Hacker-Ethik, die „Hands-on-Imperative“. Nicht nur, daß Hacker nicht an proprietärer Software basteln und neue Ideen entwickeln konnten – diese Programme beleidigten ihr Selbstverständnis als mündige Akteure, die Probleme selbst lösten. In größerem Rahmen machten proprietäre Programme auch klar, daß es keine über das MIT hinausreichende Hacker-Solidarität gab.
- Die meisten Hacker am AI Lab arbeiteten Anfang der 80er nebenbei für mehrere große IT-Firmen. Das Vordringen vor allem einer dieser Firmen auf den Campus führte dazu, daß die Studenten ihre Kenntnisse untereinander nicht mehr austauschen konnten, da sie in unterschiedlichem Maße zu Geheimhaltung verpflichtet waren. Es bildeten sich Gruppen nach Firmenzugehörigkeit. Dadurch wurde der zweite Grundsatz der Hacker-Ethik gebrochen: „All information should be free“.

Der Bruch dieser drei Regeln verursachte jeweils heftige Konflikte – ausgetragen wurden sie vor allem von Richard Matthew Stallman, der Anfang der 80er als knapp Dreißigjähriger eine wichtige Rolle[170] innerhalb der Hacker-Community am MIT spielte. Wegen seines lautstarken und idealistischen Widerstands bezeichnete LEVY Stallman als „last true hacker“[171], freilich ohne zur Zeit der Niederschrift im Jahr 1984 zu ahnen, daß dieser Widerstand eine neue Generation von Hackern ins Leben rufen sollte.

Die Berichte darüber, wie Stallman gegen jede der genannten drei Verletzungen der Hacker-Ethik ankämpfte, bilden inzwischen die drei „Gründungslegenden“ der Free Software-Bewegung. Im folgenden sollen diese Legenden kurz erzählt werden, nach einer kurzen Einführung seiner Person.

---

[170] Vgl. RAYMOND: The Cathedral …, S. 11.

[171] LEVY, a.a.O., 435.

Abbildung 13: Richard M. Stallman am 26.09.2004 bei einer Veranstaltung des FFII gegen Softwarepatente in Frankfurt am Main. *Foto: Ulrich Hansen*

Richard Matthew Stallman hatte 1970 im Alter von 18 Jahren ein Studium der Mathematik und Physik an der Harvard-Universität begonnen.[172] Nach knapp einem Jahr lernte er das AI Lab am zwei Kilometer entfernten MIT kennen und wurde dort 1971 als Systemprogrammierer beschäftigt – das übliche formale Beschäftigungsverhältnis der dortigen Hacker. 1974 beendete er sein Physik-Studium mit Magna cum laude und begann als Graduierter am MIT. 1975 warf er dieses Vorhaben hin und widmete sich ausschließlich dem Programmieren. Zu seinen Arbeiten am AI Lab zählen die Weiterentwicklung des bereits erwähnten Incompatible Time-sharing Systems ITS, die Erstellung eines Programms zur automatisierten Analyse von Schaltkreis-Diagrammen sowie des programmierbaren Texteditors Emacs[173].

Bemerkenswert an letzterem Programm, Emacs, ist, daß Stallman sich nach Fertigstellung des Editors im Jahr 1979 für eine besondere Form des Copyrights entschieden hatte. Eine völlige Freigabe kam nicht in Frage – was daraus entstehen konnte, hatte er am Vorgänger TECO gesehen: Individuelle Verbesserungen führten dazu, daß an allen Arbeitsplätzen im AI Lab unterschiedliche Versionen von TECO liefen, die jeweils eine Einarbeitung erforderten. WILLIAMS spricht in diesem Zusammenhang vom „Tower of Babel effect“[174]. Ein Verbot solcher Veränderungen jedoch – durch Ausüben des Copyrights und z. B. der Geheimhaltung des Quellcodes – lehnte Stallman aus Gründen der Hacker-Ethik

---

[172] Vgl. MOODY, a.a.O.,. 15.

[173] Emacs erlangte später aus anderem Grund einige Berühmtheit: Die 1986 verteilte Fassung des Programms, GNU Emacs, enthielt einen Fehler, der es erlaubte, unrechtmäßig Administratorrechte auf einem Unix-Rechner zu erlangen. Genau diesen Angriffspunkt nutzten damals Cracker aus Hannover, um im Auftrag des KGB mehrere Rechner des US-Militärs anzugreifen. Vgl. STOLL, Clifford: Das Kuckucksei. Die Jagd auf die deutschen Hacker, die das Pentagon knackten. Frankfurt am Main: Fischer, 1990, S. 35f.

[174] WILLIAMS, a.a.O., S. 84.

ebenfalls ab.[175] Statt dessen verfügte er: “It [Emacs] is distributed on a basis of communal sharing, which means that all improvements must be given back to me to be incorporated and distributed.”[176]

Emacs war damit eines der ersten community-basierten Programme, wobei Stallman allerdings – im Vergleich zu heutigen Projekten – eine ungewöhnlich herausgehobene Position einnahm.[177] Die beschriebene Vertriebsbestimmung für Emacs war das Vorbild für das Copyleft-Konzept und zeigt, daß Stallmans Grundideen zeitlich bereits vor den nachfolgend geschilderten Konflikten bzw. „Gründungslegenden“ vorhanden waren.

2.2.1 Erste Legende: Der Kampf gegen Paßwörter

1982 war das paßwortlose Hacker-Betriebssystem ITS passé, da die bisherige Hardwareplattform, die PDP-10, nicht weiterentwickelt wurde und am AI Lab inzwischen die Zahl und die Zeit der Hacker für eine Anpassung von ITS an eine neue Hardware zu gering war. Als Ersatz wurde der Einsatz des kommerziellen DEC TOPS-20-Sytems (Spitzname: Twenex) beschlossen, das mit modernen Sicherheitsmaßnahmen wie Dateizugriffsrechten und Paßwörtern ausgestattet war. Der Widerstand der Hacker scheiterte an der Mehrheit der Fakultätsmitglieder.[178] Als Begründung für deren Wunsch nach solchen Sicherheitsmaßnahmen führt Stallman auch die zunehmende allgemeine Erfahrung mit Computersystemen an; der Umgang damit war kein Privileg der Hacker mehr:

> “’The outside world is pushing in. [...] More and more people come in having used other computer systems. Elsewhere it is taken for granted that if anybody else can modify your files you’ll be unable to do anything, you’ll be sabotaged every five minutes. Fewer and fewer people are around who grew up here the old way, and know that it’s possible, and it’s a reasonable way to live.’”[179]

Stallman empfand diese Sicherheitsmaßnahmen auch als persönliche Provokation: Schon Ende der 70er hatte er am benachbarten Labor für Computerwissenschaften die Einführung von Paßwörtern auf Hacker-Art bekämpft: Er baute in das dortige System eine Programmschleife ein, die jeden, der versuchte, ein persönliches Paßwort zu generieren, in einer Bildschirmnachricht darum bat, statt dessen einfach die Enter-Taste zu betätigen (al-

---

[175] Stallman war schon damals ein entschiedener Gegner dieser Praxis. Als warnendes Beispiel galt ihm der Programmierer Brian Reid, der seinen Texteditor Scribe im gleichen Jahr als kommerzielle Version veröffentlicht hatte – um eine Bezahlung zu erzwingen, beendete sich das Programm nach einer bestimmten Zahl von Startvorgängen, wenn es nicht zuvor mit einem kostenpflichtigen Schlüssel freigeschaltet worden war. Stallman erklärte dazu in einem Interview: „The problem was nobody censured or punished this student for what he did. The result was other people got tempted to follow his example.“ Ebd., S. 88.

[176] STALLMAN, Richard Matthew: EMACS: The Extensible, Customizable Display Editor. Online veröffentlicht unter:
http://www.gnu.org/software/emacs/emacs-paper.html [Aufgerufen am: 22.11.2004].

[177] Stallman erklärte später, die Forderung, daß er persönlich alle Änderungen erhalten müsse, sei überzogen gewesen: „‘It was wrong to require them to be sent to one privileged developer. That kind of centralization and privilege for one was not consistent with a society in which all had equal rights.’“ WILLIAMS, a.a.O., S. 127.

[178] Ebd., S. 92.

[179] LEVY, a.a.O., S. 418f.

so ein leeres Paßwort zu benutzen): „It's much easier to type, and also it stands up to the principle that there should be no passwords."[180]

Als es Stallman 1982 auch durch Sabotageversuche[181] nicht gelang, die Einführung der Paßwörter am AI Lab zu verhindern, unterminierte er deren Sinn, indem er sein eigenes Paßwort „RMS" (nach seinen Initialen) öffentlich bekanntmachte.[182] Auch diese Praxis ging zurück auf das Vorgängersystem ITS, bei dem es frei gestattet war, sich – auch von außerhalb – unter fremden Namen anzumelden; den anonymen „Touristen" wurde sogar erlaubt, Betriebssysteminterna zu verändern.[183] Weil das (sehr viel fragilere) Twenex zur einzigen Verbindung des AI Lab zum Arpanet gemacht werden sollte, verbot man später Stallman die Veröffentlichung seines Paßworts – angeblich wurde seinetwegen sogar die gesamte Anbindung zum Arpanet in Frage gestellt.[184] Nach eigenen Angaben boykottierte er schließlich Twenex und faßte aufgrund der Erfahrungen den Entschluß zum GNU-Projekt.[185]

---

[180] WILLIAMS, a.a.O., S. 53.

[181] So gelang es ihm, durch einen Einbruch in das TOPS-20-System die verwendeten Paßwörter zu finden und in einer anonymen Botschaft zu veröffentlichen. Vgl. STALLMAN, Richard M.: Lecture at Kungliga Tekniska Högskolan (KTH, Königliches Institut für Technologie). Stockholm: 1986. Veröffentlicht online unter: http://www.gnu.org/philosophy/stallman-kth.html [Aufgerufen am 22.11.2004].

[182] Aufgrund der Anbindung des AI Lab mit dem Arpanet ermöglichte Stallman so jedem anderen im Netzwerk, den Computer am AI Lab zu benutzen: Einer dieser „Touristen" erklärte dazu später: „'I'm eternally grateful that MIT let me and many other people use their computers for free,' said Hopkins. 'It meant a lot to many people.'" WILLIAMS, a.a.O., S. 94.

[183] STALLMAN beschreibt dies als „[...] accepted tradition specially when the Arpanet began and people started connecting to our machines from all over the country. Now what we'd hope for was that these people would actually learn to program and they would start changing the operating system. If you say this to the system manager anywhere else he'd be horrified. If you'd suggest that any outsider might use the machine, he'll say 'But what if he starts changing our system programs?' But for us, when an outsider started to change the system programs, that meant he was showing a real interest in becoming a contributing member of the community. We would always encourage them to do this." STALLMAN: Lecture at Kungliga Tekniska Högskolan (KTH, Königliches Institut für Technologie), a.a.O.
Sehr komisch wird die Konfrontation von konventionellem Sicherheitsdenken mit der prinzipiellen Offenheit am AI Lab, wenn ein nichtsahnender deutscher Cracker versucht, in ITS einzubrechen, wie von STOLL beschrieben: „Weil die KI-Forschung so abgehoben ist, fand er nicht viel. Natürlich bietet das antike Betriebssystem nicht viel Sicherheit – jeder Benutzer konnte die Dateien eines jeden anderen lesen. Aber der Hacker merkte das nicht. Die reine Unfähigkeit, ihr System zu verstehen, schützte ihre Information." STOLL, a.a.O., S. 125.

[184] GRASSMUCK sieht sogar das Pentagon involviert: Stallman habe seinen Kampf gegen Paßwörter so erfolgreich betrieben „bis schließlich das US-Verteidigungsministerium drohte, das KI-Lab vom ARPAnet abzuhängen. In einem zunehmend wichtigeren Netz ging es nicht an, daß jedermann, ohne eine Legitimation vorzuweisen, durch diese weit offene Tür spazieren und sich in militärischen Rechnern tummeln konnte." GRASSMUCK, Volker: Freie Software. Zwischen Privat- und Gemeineigentum. Bonn: Bundeszentrale für politische Bildung, 2002, S. 219. Meiner Ansicht nach ist diese Darstellung arg übertrieben: Das MIT war personell so eng verflochten mit ARPA (z.B. durch J. C. R. Licklider und Robert Fano), daß es keiner förmlichen Drohung der ARPA oder gar des Pentagons selbst bedurfte.

[185] „It was clear to me from what I saw happening to these proprietary systems that the only way we could have the spirit of the old AI lab was to have a free operating system. To have a system made up of free software which could be shared with anyone. So that we could invite everyone to join in improving it. And that's what led up to the GNU project." STALLMAN, Richard M.: Lecture at Kungliga Tekniska Högskolan (KTH, Königliches Institut für Technologie). a.a.O.

2.2.2 Zweite Legende: Der Ärger über einen Druckertreiber

Ein hartnäckig zitiertes[186] Motiv Stallmans für den Start des GNU-Projekts ist auch sein Ärger über einen kommerziellen Druckertreiber. WILLIAMS zufolge hatte das AI Lab im Jahr 1980 für sein Netzwerk einen der ersten Laserdrucker angeschafft, ein Modell der Firma Xerox, das allerdings die Eigenschaft besaß, häufig einen Papierstau zu produzieren. Dies führte zu Frustration bei den Benutzern, die den Fehler erst bemerkten, wenn sie zum Gerät kamen, um ihre Ausdrucke abzuholen. Beim vorherigen Modell hatte Stallman als Administrator einen simplen Regelkreislauf erstellt: Das Computersystem prüfte den Drucker alle paar Minuten und benachrichtigte im Falle eines Staus alle Nutzer, die gerade druckten, mit den Worten „The printer is jammed, please fix it“[187].

Eine solche Lösung war mit dem neuen Modell nicht möglich, entdeckte Stallman. Die Treiber, also die Programme, um den Drucker anzusteuern, lagen lediglich in nicht anpaßbarer kompilierter Form vor, nicht aber im (lesbaren) Quellcode.[188] Ein Xerox-Mitarbeiter, den Stallman nach einigem Zögern um den Code bat, verweigerte diesen mit Hinweis auf ein „non-disclosure agreement“, also eine Schweigepflicht. Stallman reagierte wütend („Maybe I slammed the door“[189]) auf diese Verletzung der ersten Regel der Hacker-Ethik; er betrachtete den Xerox-Drucker als eine Art trojanisches Pferd:

> “It was my first encounter with a nondisclosure agreement and it immediately taught me that nondisclosure agreements have victims. […] In this case I was the victim. […] From that day forward, I decided this was something I could never participate in. […] I decided never to make other people victims just like I had been a victim.”[190]

Stallman zufolge bestärkte ihn diese Erfahrung darin, seine (zu diesem Zeitpunkt vagen) Pläne weiterzuentwickeln, wie Software geteilt werden könnte.[191]

2.2.3 Dritte Legende: Der "Symbolics-Krieg"

Seit 1975 hatte man am AI Lab an der „Lisp Machine“ gearbeitet. Es handelte sich dabei um einen Computer, der speziell für Anforderungen der Programmiersprache Lisp (LiSt Processing Language) konstruiert ist. Lisp und die Lisp Machine erwiesen sich später als wichtige Instrumente zur Programmierung sogenannter Expertensysteme. Auch an diesem Computer war das US-Verteidigungsministerium interessiert: ARPA bestellte Ende der 70er mindestens sechs dieser Rechner für jeweils 50.000 Dollar.[192]

---

[186] Eine Google-Suche ergibt rund 160 Einträge für „Stallman“ und „Druckertreiber“, also allein für die deutsche Version der Legende.

[187] WILLIAMS, a.a.O., S. 3f. Die Eleganz dieser Lösung lag Stallman zufolge darin, daß die Meldung nur an die Nutzer erging, die ein dringendes Bedürfnis nach einem funktionierenden Drucker hatten sowie darin, daß niemandem versprochen wurde, daß einem ein anderer die Arbeit abnehmen würde.

[188] Für eine Erklärung des Begriffs Quellcode siehe das Kapitel „Das Schwein guckt ins Uhrwerk oder: Was ist Quellcode?“ im Anhang dieser Arbeit.

[189] STALLMAN, Richard Matthew: Free Software: Freedom and Cooperation. In Ders.: Free Software, Free Society. Selected Essays. Boston: GNU-Press, 2002, S. 155-186, hier S. 158.

[190] WILLIAMS, a.a.O., S. 11f.

[191] Vgl. ebd., S. 10.

[192] Vgl. LEVY, a.a.O., S. 421.

Um diese offensichtlichen Marktchancen auszunutzen, versuchte der Erfinder der Lisp Machine, der MIT-Dozent und Programmierer Richard D. Greenblatt[193] einen früheren Administrator der Universität, Russel Noftsker, zur Gründung einer Firma zu gewinnen. Da sich beide aber über die Organisation nicht einigen konnten – Greenblatt lehnte fremde Kapitalgeber ab – wurden im Jahr 1980 zwei Firmen gegründet: ***Symbolics*** (von Noftsker) und ***Lisp Machines Incorporated*** (LMI, von Greenblatt).

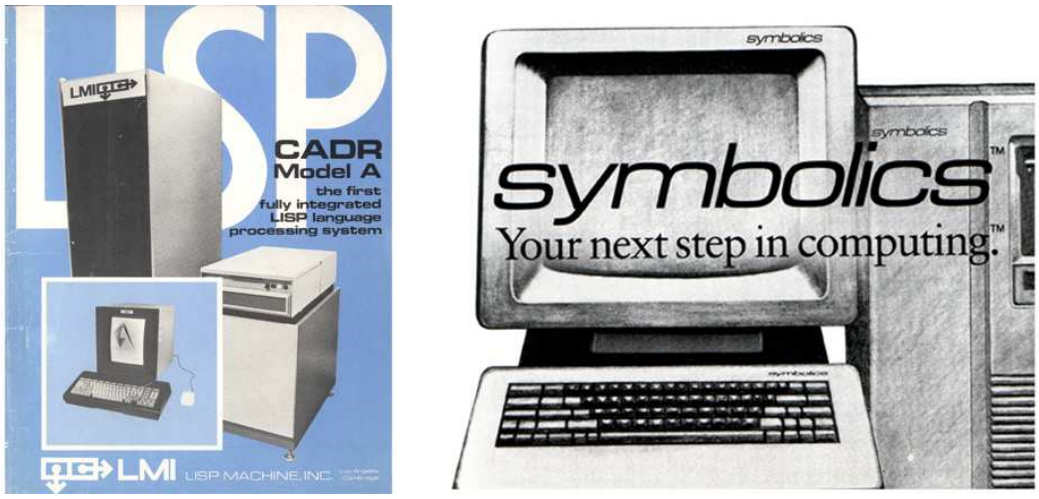


Abbildung 14: Ausriß aus Werbebroschüren der Firmen LMI und Symbolics.[194]

Beide Spin-Offs rekrutierten ihre Mitarbeiter unter den Hackern am AI Lab,[195] die teils schon jahrelang an dem System gearbeitet hatten. Ihr Arbeitsplatz blieb der Gleiche: Greenblatt arbeitete ohnehin im Universitätsgebäude; Symbolics besaß zunächst keine eigenen Räume, so daß man zu der bemerkenswerten Übereinkunft kam, daß beide konkurrierende Firmen im gleichen Raum mit der gleichen Ausstattung arbeiteten. Als Ausgangsbasis verwendete man sogar die gleiche Software, Lisp Machine OS, das vom AI Lab entwickelte Lisp-Betriebssystem.

Wie zu erwarten war, führte diese Vermischung zu erheblichen sozialen Spannungen. Dank einer professionelleren Organisation und Finanzierung gelang es Symbolics, einen Großteil der Hacker abzuwerben, während Greenblatts LMI auf weit weniger Enthusiasmus stieß.[196] Im Laufe der Zeit nutzte Symbolics seine stärkere Position, um LMI zu isolieren: So wurde das MIT erfolgreich aufgefordert, das Anstellungsverhältnis mit allen Mitarbeitern von LMI (inklusive Greenblatt) zu lösen, da es als öffentliche Einrichtung keinen privaten Wettbewerber zu unterstützen habe.[197] Mitarbeiter beider Firmen durften

---

193 Dabei handelte es sich um denselben Greenblatt, der 1969 eines der ersten leistungsfähigen Schachprogramme entwickelt hatte. LEVY ist von seinen Fähigkeiten so beeindruckt, dass er ihn nur als „the hackers hacker“ bezeichnet. Ebd., S. 10.

194 Die Abbildung links zeigt die Broschüre zur ersten kommerziellen Lisp Machine von LMI, der LMI-CADR aus dem Jahr 1980.
Quelle: http://www.bitsavers.org/pdf/mit/cadr/CADR_brochure1.jpg [Aufgerufen am 10.10.2005].
Die Abbildung rechts zeigt das Logo der Firma Symbolics. Quelle:
http://www.cs.northwestern.edu/~hunicke/blog/images/ai85/symbolics.jpg [Aufgerufen am 10.10.2005].

195 14 Hacker am AI Lab wurden von Symbolics als Teilzeitkräfte angeworben und bezahlt, alle übrigen arbeiteten für Greenblatt. Vgl. WILLIAMS, a.a.O., S. 95. WILLIAMS verweist in diesem Zusammenhang auf: NEWQUIST, H. P.: The Brain Makers: Genius, Ego, and Greed in the Quest for Machines that Think. Indianapolis: Sams Publishing, 1994, S. 172.

196 Vgl. LEVY, a.a.O., S. 423f.

197 Vgl. ebd., S. 424.

am MIT nicht mehr über ihre Arbeitsinhalte sprechen[198] und schließlich untersagte es Symbolics der Universität sogar – entgegen früherer Zusagen[199] – Softwareänderungen von Symbolics weiter in die institutseigenen Lisp-Machine einzupflegen, da man befürchtete, daß LMI diese Änderungen kopieren könnte.[200]

Dieses Verbot betraf Richard M. Stallman: Er war zu dieser Zeit verantwortlich für die Lisp Machine des MIT, auf der er die Verbesserungen beider Firmen installierte. Zwar wurde er von keiner der beiden Parteien bezahlt; man darf aber annehmen, daß er Symbolics nicht freundlich gesonnen war, da er die Spaltung der Hacker-Community bereits zuvor kritisiert hatte. Das Verbot bezeichnete er als „Kriegserklärung".[201] Daß Symbolics es der Universität verbieten wollte, ein System produktiv zu nutzen, das von Hackern am AI Lab begonnen worden war und das unter Spaltung der Hacker-Community von den gleichen Hackern beider Seiten weiterentwickelt worden war, war für ihn die groteske Schlußpointe einer ohnehin bitteren „Zusammenarbeit".

Als Konsequenz daraus bezog Stallman offen Stellung gegen Symbolics und zugunsten von LMI und setzte dabei seine Programmierfähigkeiten ein: In dem fast zwei Jahre lang andauernden sogenannten „Symbolics War"[202] begann Stallman, alle neuen Programmfunktionen des Symbolics-Systems, von denen das MIT nach wie vor Kenntnis erhielt, für Greenblatts Firma LMI nachzubilden.

> "So, in an effort to help keep Lisp Machines going, I began duplicating all of the improvements Symbolics had made to the Lisp machine system. I wrote the equivalent improvements again myself (i.e., the code was my own). [...] When they made a beta announcement that gave the release notes, I would see what the features were and then implement them. By the time they had a real release, I did too. In this way, for two years, I prevented them from wiping out Lisp Machines Incorporated, and the two companies went on."[203]

---

198 Die Geheimhaltungsvorschriften führten zu der paradoxen Situation, daß die langjährigen Hacker-Kollgen Bill Gosper und Richard Greenblatt nicht mehr über ihre Arbeitsinhalte reden durften. Vgl. LEVY, a.a.O., S. 424.

199 WILLIAMS zitiert Stallman mit den Worten, es habe ein „Gentleman's Agreement" zwischen MIT und Symbolics darüber gegeben, daß Weiterentwicklungen von Symbolics auf dem MIT-Computer installiert werden durften. Symbolics habe diese Vereinbarung am 16. März 1982 beendet. WILLIAMS, a.a.O., S. 95. Später spricht Stallman von einem Vertrag, den Symbolics, LMI und das MIT abgeschlossen hätten; danach habe das MIT Code von Symbolics zwar allgemein verwenden dürfen, aufgrund eines unpräzisen Vertragstextes habe Symbolics aber argumentieren können, dies schließe nicht den Einsatz auf Computern des MIT ein. Vgl. STALLMAN, Richard Matthew: My Lisp Experiences and the Development of GNU Emacs. Mitschrift einer Rede Stallmans auf der „International Lisp Conference" am 28.10.2002. http://www.gnu.org/gnu/rms-lisp.html [Aufgerufen am 02.12.2004].

200 Vgl. WILLIAMS, a.a.O., S. 96. Wikipedia verweist dagegen auf einen Symbolics-Mitarbeiter, der Stallman selbst die Schuld für den Entschluß ihrer Firma gibt, den Sourcecode nicht weiter zur Verfügung zu stellen: „Until 1981, they shared all the source code with MIT and kept it on an MIT server. The reason for the change in policy, according to a Symbolics employee, was due to Richard Stallman making changes that they disagreed with, and at one point leaving the software in a state where it wouldn't compile. Stallman's account of the reason for the split has never been published. Stallman had been excluded from the group of engineers and managers that founded Symbolics although all had worked at MIT together."
WIKIPEDIA (engl.): Symbolics. In: http://en.wikipedia.org/wiki/Symbolics [Aufgerufen am 01.12.2004].

201 „'The way I saw it, the AI Lab was a neutral country, like Belgium in World War I,' Stallman says. 'If Germany invades Belgium, Belgium declares war on Germany and sides with Britain and France.'" WILLIAMS, a.a.O., S. 96.

202 Ebd., S. 96.

203 STALLMAN: My Lisp …, a.a.O.

Von März 1982 bis Ende 1983 programmierte Stallman am AI Lab ganz alleine alles nach, womit sich bei Symbolics eine ganze Gruppe von Programmierern beschäftigte, eine Leistung, die bei Symbolics-Mitarbeitern Ärger, aber auch Bewunderung auslöste: „'Stallman doesn't have anybody to argue with all night over there. He's working alone! It's incredible anyone could do this alone!'"[204]

Stallman gab Ende 1983 auf. Es war ihm nicht gelungen, den Erfolg der Gegenseite zu verhindern. Mit dem Verkauf der zwischen 36.000 und 125.000 Dollar[205] teuren Lisp Machines erzielte Symbolics 1984 einen Umsatz von mehr als 40 Millionen Dollar; bis 1986 stieg dieser Umsatz sogar auf 114,2 Millionen Dollar.[206] LMI konnte trotz einer Zusammenarbeit mit Texas Instruments nicht an diesen Erfolg anschließen und meldete 1987 Bankrott an. Tröstlich für Anhänger Stallmans ist allerdings, daß auch Symbolics' Erfolg ein Strohfeuer blieb[207] und mit der Einstellung der Star-Wars-Pläne[208] von US-Präsident Reagan weitgehend endete. Zwar war der Firma noch ein kurzes Überleben mit dem Verkauf von Grafik-Computern vergönnt, heute jedoch beschränkt sich ihre Tätigkeit auf die Wartung noch existierender Systeme, vor allem von Software.

In Erinnerung bleibt die Firma dennoch – und das nicht nur wegen ihrer Rolle bei der Entwicklung von GNU: Symbolics beantragte und erhielt im März 1985 die erste jemals registrierte Internet-Domain.

2.3 Das GNU-Projekt

Ende 1983 war die Hacker-Gemeinschaft am MIT zerstört. Stallman selbst war beruflich und persönlich isoliert. Während des Symbolics-Konflikts war er von gemeinsamen Aktivitäten anderer Hacker – wie dem traditionellen Essen im chinesischen Restaurant – ausgeschlossen worden, möglicherweise auf Druck von Symbolics,[209] möglicherweise aber

---

[204] LEVY zitiert hier den MIT-Hacker und Symbolics-Mitarbeiter Bill Gosper. LEVY, a.a.O., S. 424.

[205] NEWQUIST, Harvey P.: AI Trends '88. A comprehensive Annual Report of the AI Industry. DM Data, 1988. Zitiert nach: GRAYLIN, Alvin; KJOLAAS, Kari Anne Hoier; LOFLIN, Jonathan; WALKER, Jimmie Dale: Symbolics, Inc. A failure of heterogeneous engineering. Cambridge, Massachusetts: 1998. Online unter: http://mit.edu/6.933/www/Symbolics.pdf [Aufgerufen am 25.11.2004], S. 32. Bei der Studie handelt es sich um das Ergebnis eines studentischen Projekts, das im Herbst 1998 im Rahmen des Seminars „The Structure of Engineering Revolutions" am MIT durchgeführt wurde.

[206] Ebd., S. 4.

[207] Bereits 1987 machte die Firma einen Verlust von mehr als 20 Millionen Dollar und kam bis zum Bankrott 1993 nicht mehr aus den roten Zahlen. Vgl. ebd., S. 4.

[208] Allerdings war das US-Verteidigungsministerium nur indirekt involviert: Viele Käufer von Lisp Machines entwickelten darauf Expertensysteme, die aus Mitteln des SDI-Programms gefördert wurden. Als das Programm eingestellt wurde, nahm auch das Interesse an der Lisp Machine ab. Vgl. ebd., S. 30. Dort heißt es: " SDI funding did not go directly to Symbolics. However, SDI funds did go to many of Symbolics' customers who were developing expertsystems. The demand for Lisp machines in the early 1980s was largely fueled by this phenomenon."

[209] WILLIAMS, a.a.O., S. 98.

auch wegen des eigenen Verhaltens.[210] WILLIAMS deutet an, daß der 31jährige unter Studenten als Anachronismus galt.[211] Er selbst sah sich damals ebenso:

> "'The AI lab used to be the one example that showed it was possible to have an institution that was anarchistic and very great,' he would explain. [...] 'For a while we were setting an example for the rest of the world. Now that this is gone, where am I going to begin from? I read a book the other day. It's called Ishi, the Last Yahi. It's a book about the last survivor of a tribe of Indians, initially with his family, and then gradually they died out one by one.' [...] 'I'm the last survivor of a dead culture,' said RMS. 'And I don't really belong in the world anymore. And in some ways I feel I ought to be dead.'"[212]

WILLIAMS vermutet deshalb neben ideologischen Gründen für den Start des GNU-Projekts auch ein persönliches Motiv Stallmans: „Crushing loneliness"[213] – das Zermalmen von Einsamkeit.

Stallman gab proprietärer Software die Schuld an dem Dilemma, in dem er sich befand: Hätte er nachgegeben und sich an der Entwicklung solcher Software beteiligt, hätte er zwar nach eigenen Angaben seine berufliche Situation verbessern können, aus moralischer Sicht aber hätte er sein Leben damit verbracht, „Mauern zu errichten, um Menschen voneinander zu isolieren"[214] – was er ablehnte. Allein dieses moralische Dilemma zeigt, welche Bindungskraft die Hacker-Ethik auf jemanden ausübte, der dreizehn Jahre lang Teil dieser Gemeinschaft gewesen war (und der sich selbst ansonsten als Atheisten bezeichnet). Stallmans Lösung war so logisch, wie vermessen: "The problem, the dilemma existed for me and for everyone else because all of the available operating systems for modern computers were proprietary. [...] So I realized that there was something I could do that would solve the problem. I had just the right skills to be able to do it. [...] So I felt, ' I'm elected. I have to work on this. If not me, who?'"[215]

Bereits zuvor hatte Stallman mit dem Gedanken gespielt, ein freies Betriebssystem zu schreiben, das das populäre Unix-System nachbilden sollte. Schriftlich angedeutet hatte er diese Pläne unter anderem in Computerhandbüchern (sic!). So schreibt er im März 1983 im Handbuch des E-Mailprogrammes ZMail:

> "The author of this manual believes that the commercialization of the computer industry hinders the further development of systems such as decribed herein. He considers proprietary software morally objectionable and plans to dedicate his career to promoting the sharing and free exchange of software."[216]

---

[210] Stallmans schroffe Art auch gegen offensichtliche Bewunderer und seine kompromißlose Form der Diskussion ist heute noch bei Vorträgen zu erleben und kontrastieren mit dem ihm gleichfalls eigenen Witz und Charisma. Vgl.: HANSEN, Ulrich: Hackerlegende gegen Softwarepatente. „Solange wir kämpfen, können wir gewinnen" – Richard M. Stallman auf Blitzbesuch in Europa. ZDF heute-online vom 27.09.2004. http://www.heute.t-online.de/ZDFheute/artikel/29/0,1367,COMP-0-2197021,00.html [Aufgerufen am 23.11.2004].

[211] WILLIAMS, a.a.O., S. 97.

[212] LEVY, a.a.O., S.427.

[213] WILLIAMS, a.a.O., S. 198.

[214] „I'd have to look back at my career and say, ' I've spent my life building walls to divide people,' and I would have been ashamed of my life." STALLMAN: Free Software: Freedom…, a.a.O., S. 159f.

[215] Ebd., S. 160.

[216] STALLMAN, Richard Matthew: LISP Machine ZMail Manual. First Edition, ZMail Version 5.0 (System 94). Boston: MIT, März 1983. S. 1. Archiviert online unter: http://www.bitsavers.org/pdf/mit/cadr/LISPM_ZmailMan_Apr83.pdf [Aufgerufen am 10.10.2005].

Im August 1983 wird er bereits präziser: In dem Handbuch für das Lisp Window-System:

> "Starting in December 1983 I plan to work on the development of GNU, a complete Unix-compatible software system for standard hardware architectures, to be shared freely with everyone just like EMACS. This will enable people to use computers without agreeing to the idea of proprietary software. This project has inspired a growing movement of enthusiastic supporters. If you like to join it, write to me at the address of the previous page. Help get programmers sharing again. Contributions of part-time programming help will be very welcome, as will funding from philanthropists to support full-time workers, and donations or loans of computers." [217]

Diese Art der Veröffentlichung (im Fall des Lisp Window-Manuals sogar in abgesetzter Form vom Rest des Handbuchs) erreichte aber offenbar nicht genügend Enthusiasten. Einen Monat später folgt deshalb die offizielle Ankündigung, unübersehbar für jeden, der sich damals mit UNIX-Systemen beschäftigte: Am 27. September 1983 schreibt Stallman in einer E-Mail an die Internet-Foren „net.unix-wizards" und „net.usoft":

> "Free Unix!
> Starting this Thanksgiving I am going to write a complete Unix-compatible software system called GNU (for Gnu's Not Unix), and give it away free to everyone who can use it. Contributions of time, money, programs and equipment are greatly needed. [...]
>
> Why I Must Write GNU
> I consider that the golden rule requires that if I like a program I must share it with other people who like it. 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."[218]

Der Name GNU stand für „GNU's Not Unix" und bildete zum einen ein rückbezügliches Akronym – ein geläufiger Scherz am MIT[219] – zum anderen bezeichnete es das Wappentier[220] der Software, ein Gnu.

Abbildung 15: Das Logo des GNU-Projekts[221]

---

[217] STALLMAN, Richard Matthew; WEINREB, Daniel; MOON, David: Lisp Machine Window Manual. Edition 1.1. System Version 95. Boston: MIT, August 1983. S. 2. Archiviert online unter: http://www.bitsavers.org/pdf/mit/cadr/LISPM_WindowSystem_Aug83.pdf [Aufgerufen am 10.10.2005].

[218] STALLMAN, Richard Matthew: New UNIX implementation. E-Mail vom 27.09.1983, 12:35:59 Uhr an die Newsgroup „net.unix-wizards.net.usoft". Archiviert unter: http://www.gnu.org/gnu/initial-announcement.html [Aufgerufen am 04.12.2004].

[219] Andere Beispiele dafür sind: Tint (Tint Is Not TECO), Fine (Fine is not Emacs), Sine (Sine is not Emacs), Eine (Eine is not Emacs), Zwei (Zwei was Eine initially), MINCE (Mince is not Complete Emacs).

[220] Viele freie Programme tragen ein Tierwappen: Neben dem bekannten Pinguin für den Linux-Kernel gibt es zum Beispiel den MySQL-Delphin, den Firefox-Katzenbär, den Freedos-Fisch, das Perl-Kamel, den PostgreSQL-Elefanten, die Python-Schlange und den Mozilla-Drachen.

Die Entscheidung, daß sein Programm ein Unix-Klon sein sollte, traf Stallman nicht aus Gründen des Designs sondern aus Pragmatismus: Unix sei nicht sein Idealsystem, heißt es in seinem „GNU Manifesto“[222]; es habe aber den Vorteil, daß es auf verschiedene Plattformen portierbar sei und schon zahlreiche Nutzer habe.[223] Die Entscheidung erwies sich als äußerst glücklich, wie ein kurzer Exkurs zur Bedeutung von Unix deutlich macht.

2.3.1 Die Bedeutung von Unix

Die Wurzeln von Unix führen zurück bis zum 1961 am MIT begonnene Project MAC (dessen Finanzierung durch das US-Verteidigungsministerium ja die Grundlage der ersten Hacker-Gemeinschaft gebildet hatte[224]). Das Militär hielt damals die Entwicklung von mehrbenutzerfähigen („time-sharing“) Betriebssystemen für wichtig, um die landesweit knappen und teuren Computerressourcen besser auszunutzen.[225] Die Arbeiten am Project MAC mündeten am MIT in die Entwicklung der Betriebssysteme *„Compatible Time-sharing System“* (*CTSS)* ab 1961[226] und *Multiplexed Information and Computing System (Multics)* ab 1964. Letzteres war vom MIT gemeinsam mit General Electric (später Honeywell) und den AT&T Bell Labs entworfen worden. Der Widerstand der Hacker – also der wichtigsten Nutzer und Administratoren – verhinderte aber den Einsatz beider Betriebssysteme am AI Lab; als Gegenmodell zu CTSS entstand das *„Incompatible Time-sharing System“* (ITS).[227]

Als Gegenmodell zu Multics wiederum entstand *Unix*, allerdings nicht am MIT sondern bei AT&T. Aus Unzufriedenheit mit der Trägheit und dem Ressourcenhunger von Multics hatte man dort die Weiterentwicklung beendet – dabei die Rechnung aber ohne die eigenen Programmierer gemacht. Zwei von ihnen, Ken Thompson[228] und Dennis Ritchie entwarfen im Jahr 1969 ein weniger ressourcenhungriges System, das sie – als Verballhornung von Multics – zunächst „Unics“ und dann „Unix“ nannten.[229]

---

[221] In der Version von Aurélio A. Heckert; das Original wurde vermutlich gezeichnet von Etienne Suvasa. Vgl. http://www.gnu.org/graphics/agnuhead.html [Aufgerufen am 04.12.2004].

[222] STALLMAN, Richard: The GNU Manifesto. In: Ders. (Hrsg.): Free Software, Free Society. Selected Essays. Boston: GNU-Press, 2002, S. 31-39, hier S. 32.

[223] STALLMAN: Free Software: Freedom…, a.a.O., S. 161.

[224] Siehe dazu auch Kapitel 1.4

[225] Vgl. LEVY, a.a.O., S.67-69.

[226] CTSS lief auf einem IBM 7094-Computer. Bemerkenswert ist, daß es eines der ersten Systeme – möglicherweise sogar das erste System – war, das Email von einem Benutzer zum anderen ermöglichte.

[227] Siehe Abschnitt 1.1.3

[228] Obwohl Ritchie an der Konzeption von Unix beteiligt war, erbrachte Thompson die Haupt-Programmierleistung: Im August 1969, innerhalb eines Monats, schrieb er die wichtigsten Komponenten; seine Frau war in dieser Zeit zu Verwandten geflogen. Vgl. MOODY, a.a.O., S. 14.

[229] Unics stand für „Uniplexed Information and Computing System“. Weil das Wort „Unics“ im Englischen ähnlich klingt wie „Eunuchs“ (und dies zu dem Kalauer führte, Unics sei ein kastriertes Multics) versuchte man eine deutlichere Betonung zu erzwingen, indem man die Endung als „x“ schrieb. Vgl. WIKIPEDIA (engl.): Unix. In: http://en.wikipedia.org/wiki/Unix [Aufgerufen am 4.12.04].

Abbildung 16: Thompson (links) und Ritchie erhalten 1999 von US-Präsident Clinton die „National Medal of Technology“, die höchste Auszeichnung für technologische Innovationen.[230]

Nicht genug, daß diese Entwicklung allein auf die eigene Initiative von Thompson und Ritchie zurückging, sie wurde auch von Bell Labs nicht im Geringsten gefördert. Bis 1971 hielten die beiden Programmierer ihr Projekt sogar vor ihrem Arbeitgeber geheim: Nach außen hin gaben Thompson und Ritchie ihre Software als Textverarbeitung aus.

> “Management remained blissfully unaware that the word-processing system that Thompson and colleagues were building was incubating an operating system. Operating systems were not in the Bell Labs plan [...]”[231]

So bizarr sich dieses Vorgehen der beiden Programmierer heute anhört, es zeigt die hohe Motivation, mit der sie an dem System arbeiteten. Dennis Ritchie erklärt das mit der Programmierer-Gemeinschaft, die man nach dem Ende des Multics-Projektes bewahren wollte – offenbar hatte sich hier eine eigene Hacker-Kultur gebildet:

> “What we wanted to preserve was not just a good environment in which to do programming, but a system around which a fellowship could form. We knew from experi-

---

[230] Sie erhielten die Medaille für die Erfindung des Betriebssystems Unix und der Programmiersprache C. In der Begründung erklärte Clinton, Unix und C hätten die führende Rolle der USA in der Informationszeitalter gestärkt. DIETRICH, Oliver: Höchste Technologie-Auszeichnung für Unix-Erfinder. heise-Newsticker vom 15.12.1998, 00:00, http://www.heise.de/newsticker/meldung/3364 [Aufgerufen am 10.10.2005].
Quelle des Fotos: CUNNINGHAM, Donna; REGAN; Patrick: Ritchie and Thompson Receive National Medal of Technology from President Clinton. Pressemitteilung der Fa. LUCENT Technologies / Bell Labs Innovations. Online unter: http://www.bell-labs.com/news/1999/april/28/1.html [Aufgerufen am 10.10.2005].

[231] RAYMOND, Eric Steven: The Art of UNIX Programming. Boston: Addison-Wesley, 2003. Zitiert nach der Online-Veröffentlichung auf:
http://www.catb.org/~esr/writings/taoup/html/ch02s01.html [Aufgerufen am 12.12.2004].

> ence that the essence of communal computing, as supplied by remote-access, time-shared machines, is not just to type programs into a terminal instead of a keypunch, but to encourage close communication."[232]

1973 wurde das System in der von Ritchie entwickelten Programmiersprache C komplett neu geschrieben. Bis dahin war es üblich, ein Betriebssystem möglichst hardwarenah in den vom jeweiligen Prozessor abhängigen Assemblerbefehlen zu verfassen. Die Idee Thompsons und Ritchies, Unix in einer abstrakten Programmiersprache zu schreiben und dann jeweils auf die entsprechende Zielplattform hin zu kompilieren, machte Unix als erstes Betriebssystem auf verschiedenen Rechnertypen lauffähig. Diese *Portabilität* sorgte in erheblichem Maße für die Verbreitung von Unix.[233]

AT&T durfte als Telekom-Monopolist keine Software verkaufen, so daß die Firma beschloß, Hochschulen und Forschungseinrichtungen den Quellcode des Programms zur beliebigen Nutzung zu überlassen, solange insgesamt eine Lizenz dafür erworben wurde (was meist ohnehin für den Hochschulcomputer geschehen war). Dies machte aus Unix ein beliebtes Modell für die Lehre und führte zu erheblichen Weiterentwicklungen, unter denen vor allem die „Berkeley System Distribution" (BSD) hervorzuheben ist. 1982 fielen die Beschränkungen für AT&T und die Firma begann, – neben anderen[234] – Unix kommerziell zu vermarkten, führte eine restriktive Lizenzpolitik ein und veröffentlichte nicht mehr den Quellcode seines Unix, was ein wichtiger Grund für das Interesse von Wissenschaftlern und Studenten am GNU-Projekt war.[235]

Daß Unix bleibende Bedeutung erlangte, geht jedoch ein weiteres Mal auf eine Entscheidung des Pentagon und ARPA zurück: Nachdem man 1979 erfolgreich alle US-Forschungszentren durch das Arpanet[236] verbunden hatte, stellte man fest, daß die unterschiedlichen Computer dort veralteten. Beim Austausch fielen vor allem die Kosten der Netzwerksoftware ins Gewicht: Sie mußte jedesmal passend zur neuen Technik umgeschrieben werden. ARPA vereinbarte darum 1980 in einem Vertrag mit der Universität von Berkeley, daß diese das einfach zu portierende Unix zur Basis des Arpanets weiter-

---

[232] RITCHIE, Dennis M.: The Evolution of the Unix Time-sharing System. In: Lecture Notes in Computer Science #79: Language Design and Programming Methodology. Berlin: Springer-Verlag, 1980. Zitiert nach der Online-Veröffentlichung unter: http://cm.bell-labs.com/cm/cs/who/dmr/hist.pdf [Aufgerufen am 12.12.2004].

[233] Vgl.: RAYMOND, Eric Steven: A Brief History of Hackerdom. In: Ders.: The Cathedral and the Bazaar. Musings on Linux and Open Source by an Accidental Revolutionary. Sebastopol: O'Reilly, 2001, S. 1-18, hier S. 8f. Zum Unterschied zwischen Programmier- und Assemblersprache siehe den Anhang " Das Schwein guckt ins Uhrwerk oder: Was ist Quellcode?".

[234] HP, IBM und DEC brachten 1982 eigene kommerzielle Unix-Versionen heraus. AT&T trennte sich 1984 von 26 Firmen der Bell-Gruppe und durfte nun ebenfalls im Softwaregeschäft tätig sein.

[235] Zur weiteren Entwicklung von Unix siehe auch Kapitel 3.1 „Das Versagen der 'Unix-Hacker'".

[236] Bereits Anfang der 60er hatte ARPA-Bereichsleiter Licklider die Entwicklung eines Netzwerks von Computern an Hochschulen und Militäreinrichtungen angeregt; es sollte langfristig das durch einen nuklearen Angriff leicht zu störende amerikanische Telefonsystem durch ein fehlertolerantes, paketvermitteltes Kommunikationssystem ersetzen. Das Arpanet wurde bekanntermaßen später zum Internet. Vgl. LESSIG, Lawrence: The Future of Ideas: The Fate of the Commons in a Connected World. New York: Random House, 2001, S.30-34.

entwickelte.[237] Es war diese Entscheidung, die dafür sorgte, daß Unix bis heute das Betriebssystem fast aller Knotenrechner des Internets ist.[238]

2.3.2 Der Verlauf des GNU-Projekts

Die Entwicklung des GNU-Projekts verlief zunächst zäh: Stallman begann im Januar 1984 mit der Entwicklung. Er hatte zuvor als Mitarbeiter des MIT gekündigt, um zu verhindern, daß die Universität wegen des Beschäftigungsverhältnisses Anspruch auf die von ihm geschriebene Software erheben konnte. Allerdings erreichte er durch die Hilfe eines befreundeten Professors[239], daß er sein Büro im MIT mietfrei behalten durfte – die nächsten Jahre sollte er sogar darin wohnen.

Nachdem sich Stallman zunächst fruchtlos an einem Compiler versucht hatte, schrieb er als erstes Produkt des GNU-Projekts eine Portierung seines Editors Emacs. GNU Emacs war ab 1985 auf Unix-Umgebungen einsetzbar und sorgte für das finanzielle Auskommen Stallmans.[240] Als nächstes entwickelte er ein weiteres Programmier-Tool, den GNU Debugger und 1986 schließlich – im zweiten Anlauf – den GNU C Compiler GCC. Ab diesem Zeitpunkt hatten sich weitere Mitarbeiter gefunden und es entstanden zahlreiche Bestandteile von GNU wie „bash“, „bison“ und „gawk“;[241] vor allem aber hatte Stallman inzwischen aus der Emacs-Lizenz eine das gesamte Projekt abdeckende Softwarelizenz gemacht, die GNU General Public License (GPL).

2.3.3 Die GNU General Public License (GPL)

In der ursprünglichen Ankündigung des GNU-Projekts 1983 hatte es Stallman versäumt zu erklären, was er unter „Freier Software“ verstand – schlimmer noch, er hatte durch sein Versprechen, er werde GNU „frei an jeden weitergeben, der es gebrauchen kann“[242] die Saat für ein Mißverständnis gelegt, das die Free-Software-Bewegung fortan begleiten sollte, nämlich die irrige Annahme, „freie Software“ bedeute „kostenlose Software“. 1984

---

[237] Vgl. McKUSICK, Marshall Kirk: Twenty Years of Berkeley Unix. From AT&T-Owned to Freely Redistributable. In: DIBONA, Chris; OCKMAN, Sam; STONE, Mark (Hrsg.): OpenSources. Voices from the Open Source Revolution. Sebastopol: O'Reilly, 1999, S. 31-46, hier S. 35.

[238] RAYMOND zufolge war diese Entscheidung von ARPA der bedeutendste Wendepunkt in der Geschichte von Unix seit seiner Erfindung. RAYMOND: The Art…, a.a.O.

[239] Vgl. STALLMAN, Richard Matthew: The GNU Project. In Ders.: Free Software, Free…, a.a.O., S. 15-30, hier S. 19. Es handelte sich um Professor Patrick Henry Winston, der von 1972 bis 1997 dem AI Lab vorstand.

[240] Er verkaufte das Programm für jeweils 150 Dollar pro Kopie, was nicht im Widerspruch zu seinem Anspruch stand, freie Software zu schaffen: Jeder konnte (nach Erhalt des Programms) frei darüber und über den Quellcode verfügen. GNU Emacs wurde zu einem Standard-Programmierwerkzeug und viele wollten es vom Urheber selbst beziehen, so daß er von acht bis zehn Bestellungen im Monat leben konnte. Vgl. STALLMAN: Free Software: Freedom…, a.a.O., S. 162.

[241] Die „Bash“ ist eine Kommandozeilenumgebung (shell) für Unix, „bison“ ist ein Hilfsprogramm für Compiler, „gawk“ ist ein Hilfsprogramm zum Suchen und Ersetzen von Text. Vgl. WILLIAMS, a.a.O., S. 133.

[242] Im Original: „give it away free to everyone who can use it“. STALLMAN: New UNIX…, a.a.O.

holte Stallman mit dem GNU-Manifest die begriffliche Klarstellung nach.[243] Dort heißt es im Abschnitt „How GNU Will Be Available“:

> “GNU is not in the public domain. Everyone will be permitted to modify and redistribute GNU, but no distributor will be allowed to restrict its further redistribution. That is to say, proprietary modifications will not be allowed. I want to make sure that all versions of GNU remain free.”

Das GNU Manifest bezieht also die „Freiheit“ der Software auf die Art und Weise, wie man sie benutzen kann, und nicht auf den Preis. Das Recht, ein GNU-Programm zu ändern und zu verbreiten, geht einher mit der Pflicht, diese Freiheit auch weiterhin zu gewährleisten.

Mit dem Erscheinen des ersten GNU-Produkts, des Editors „GNU Emacs“ im Jahr 1985, wurden diese Rechte und Pflichten in der „GNU Emacs General Public License“[244] festgeschrieben; eine Klausel, die Stallman mit juristischer Hilfe[245] aufgesetzt hatte. Als 1989 weitere GNU-Programme fertiggestellt waren, wurde die „GNU Emacs General Public License“ in „GNU General Public License“ (GPL) umbenannt und jeder Bezug auf das Emacs-Programm entfernt: Die GPL sollte künftig alle Programme des GNU-Projekts abdecken.[246]

Um ein Programm mit der GPL zu schützen, wird den Quellcode-Dateien ein Copyright-Vermerk beigefügt sowie ein Verweis auf den vollständigen Text der GPL. Im folgenden ein Auszug aus einer Quellcode-Datei des Linux-Kernels:

```
“# This file is subject to the terms and conditions of the GNU General Public
# License. See the file “COPYING” in the main directory of this archive
# for more details.
#
# Copyright (C) 1994 by Linus Torvalds”[247]
```

Im obigen Beispiel identifiziert sich Linus Torvalds als Copyright-Inhaber und verweist auf die Bedingungen, die man erfüllen muß, um dieses Copyright einzuhalten. Die Bedingungen sind in der Datei „COPYING“ beschrieben, die den vollständigen Text der GPL enthält.

---

[243] Allerdings ist die mißverständliche Formulierung „give it away free“ auch hier vorhanden, und zwar gleich im ersten Satz. Später heißt es sogar: „Once GNU is written, everyone will be able to obtain good system software free, just like air.“ STALLMAN: The GNU Manifesto, a.a.O., S. 31 und S.34. Es mag daher der Schluß erlaubt sein, daß die Ambivalenz, die Stallman später durch Parolen wie „Free as in freedom“ bekämpfte, ursprünglich von ihm selbst ausgegangen war – daß er selbst zunächst annahm, er werde ein Betriebssystem schaffen, das grundsätzlich nichtkommerziell sein würde. Sowohl auf den Internetseiten Stallmans (gnu.org), als auch in dem oben zitierten Reader sind die entsprechenden Passagen inzwischen mit der Anmerkung Stallmans versehen, es handle sich um „achtlose“ Formulierungen – eine Verpflichtung zur kostenlosen Weitergabe sei niemals beabsichtigt worden.

[244] GNU Emacs General Public License. Online archiviert unter: http://www.free-soft.org/gpl_history/emacs_gpl.html [Aufgerufen am 09.12.2004].

[245] WILLIAMS zufolge handelte es sich dabei um den auf Copyright und Patentrecht spezialisierten Bostoner Anwalt Mark Fischer; Wikipedia zufolge war auch der New Yorker Rechtsprofessor Eben Moglen beteiligt. Vgl. WILLIAMS, a.a.O., S. 124. Und: WIKIPEDIA (englisch): GNU General Public License. Online unter: http://en.wikipedia.org/wiki/GPL [Aufgerufen am 09.12.2004].

[246] Vgl. ebd., S. 126.

[247] Auszug aus der Datei „/arch/i386/Makefile“ des Linux-Kernels 2.4.26.

2.3.3.1 Die vier Freiheiten der GPL

Die GNU GPL verfolgt das Ziel, dem Benutzer der von ihr geschützten Software die folgenden Grundfreiheiten einzuräumen, die später auch in der „Free Software Definition“ festgeschrieben wurden.[248]

Freiheit 0:[249] Die Freiheit das Programm auszuführen und zwar zu jedem erdenklichen Zweck.[250]

Freiheit 1: Die Freiheit, das Programm zu untersuchen und es an die eigenen Bedürfnisse anzupassen. (Voraussetzung dafür ist der Zugang zum Quellcode).[251]

Freiheit 2: Die Freiheit, das Programm weiterzuvertreiben und so seinem Nächsten zu helfen.[252]

Freiheit 3: Die Freiheit, das Programm weiterzuentwickeln und die eigenen Verbesserungen zu veröffentlichen, so daß sie der ganzen Gemeinschaft zugute kommen. (Auch hierfür ist der Zugang zum Quellcode Voraussetzung).[253]

Im Text der GPL selbst sind diese Freiheiten durch die ersten vier Paragraphen (also null bis drei, nach Stallmans Zählweise) geregelt.[254]

In allen vier Freiheiten lassen sich Bezüge zur Hacker-Ethik erkennen: Zwar erscheint das durch „Freiheit 0“ eingeräumte Recht, das Programm auszuführen auf den ersten Blick offensichtlich (was soll man sonst damit tun?), aber bereits der Nachsatz „zu jedem erdenklichen Zweck“ markiert einen wesentlichen Unterschied zu anderen sogenannten „Freeware“- oder „Public Domain“-Programmen, die manchmal sogar nur ein „Testen“ des Programms erlaubten und häufig auf eine „nichtkommerzielle“ Nutzung beschränkt waren.[255] Das Verbot der Diskriminierung von Anwendern und Anwendungsfeldern lehnt sich an Regel IV der Hacker-Ethik an und wurde später in der „Open Source Definition“ noch ausführlicher geregelt. Die Erlaubnis, GPL-geschützten Code „for any purpose“, also auch kommerziell zu verwenden, sollte sich später als Basis für den Erfolg der GPL herausstellen.

---

[248] Vgl. STALLMAN, Richard Matthew: Free Software Definition. a.a.O., S. 41.

[249] Die vier Freiheiten sind in den Texten der Free Software Foundation in der Zählweise von Programmierern durchnumeriert, d.h. von null bis drei. ALBERT sieht bereits in dieser Numerierung eine rechtliche Angreifbarkeit der Lizenz, denn Anwälte zählten üblicherweise ab eins. Es könne schwere Probleme nach sich ziehen, wenn man auf solche Art Anwälten unnötigen Interpretationsspielraum einräume, schreibt ALBERT und äußert damit wohl den skurrilsten Zweifel an der rechtlichen Wirksamkeit der GPL. Vgl. ALBERT, Phil: A Consumer's Review of the General Public License. In: LinuxInsider, 20.07.2004 5:45 AM PT. Online veröffentlicht unter: http://www.linuxinsider.com/story/35193.html [Aufgerufen am 04.12.2004].

[250] Im Original: „The freedom to run the program, for any purpose“. STALLMAN, Richard Matthew: Free Software Definition, a.a.O. S.41.

[251] Im Original: „The freedom to study how the program works, and adapt it to your needs. Access to the source code is a precondition for this.“ Ebd.

[252] Im Original: „The freedom to redistribute copies so you can help your neighbor“. Ebd.

[253] Im Original: „The freedom to improve the program, and release your improvements to the public, so that the whole community benefits Access to the source code is a precondition for this.“ Ebd.

[254] Ein Abdruck der vollständigen GNU GPL findet sich im Anhang dieser Arbeit.

[255] Ein Beispiel für viele dieser „selbstgemachten“ Copyrightvermerke ist Larry Walls damalige Lizenz für sein Unix-Hilfsprogramm „trn": „Copyright (c) 1985, Larry Wall. You may copy the trn kit in whole or in part as long as you don't try to make money off it, or pretend that you wrote it.“ Zitiert nach: WILLIAMS, a.a.O., S. 125. Man kann sich leicht vorstellen, welches rechtliche Chaos die Benutzung zahlreicher mit hausgemachten Lizenzen versehener Programme verursachte.

„Freiheit 1“ stellt sicher, daß der Anwender unbegrenzten Zugang zum Programm (d.h. eben auch auf dessen Quellcode) hat, daß er sich nicht den Regeln der Software beugen muß, sondern sie ändern (lassen) kann. Diese Freiheit entspricht zum einen der ersten Hacker-Regel, der „Hands-On Imperative“, die hier auf Software bezogen wird;[256] zum anderen zwingt sie den Programmierer, den Anwender nicht als Konsumenten sondern als Gleichberechtigten zu behandeln – ein Rückgriff auf die Zeiten am MIT, als Hacker Anwender und Entwickler in Personalunion waren.

Die Freiheit Nr. 2, nämlich das Weitervertreiben („redistribute“) des Programms, orientiert sich an der dritten Hacker-Regel „All information should be free“. Der Appell, „seinem Nächsten zu helfen“ („so you can help your neighbor“), mag ebenfalls aus der Erfahrung des engen persönlichen Umgangs der Hacker am MIT erklärt werden: Stallman bezeichnet die MIT-Hacker auch als „software-sharing community“[257]. Allerdings macht er mit der Formulierung „share with your neighbor“ – die in fast jeder[258] seiner Reden fällt – auch den sozialen Anspruch der Free-Software-Bewegung deutlich, und auch der religiöse Anklang ist nicht zufällig, wie er selbst erklärt:[259]

> “And, in fact, this spirit of goodwill – the spirit of helping your neighbor, voluntarily – is society’s most important resource. It makes the difference between a livable society and a dog-eat-dog jungle. Its importance has been recognized by the world’s major religions for thousands of years, and they explicitly try to encourage this attitude.”[260]

In diesem Zusammenhang ist auch der Appell an das Gemeinschaftsgefühl in der „Freiheit 3“ zu verstehen: Stallman spricht von „community“ und meint damit bereits die virtuelle Gemeinschaft der weltweit verteilten Anwender und Entwickler. Hierfür stand möglicherweise die „soziale Utopie“ der sechsten Regel der Hacker-Ethik Pate.

2.3.3.2 "Copyleft" - die Pflichten der GPL

Die oben erwähnten Freiheiten hat man auch mit einem (im Quellcode vorliegenden) Programm, das überhaupt nicht geschützt ist, dessen Urheber es also ohne Auflagen und oft sogar ohne Namensnennung als Gemeingut (engl.: „public domain“) freigegeben hat. Das geschah im studentischen und wissenschaftlichen Bereich relativ häufig.

Solche Programme, die ihrem Besitzer keinerlei Pflichten auferlegen, kann allerdings jeder nehmen, verändern und unter eigenem Namen verkaufen – ohne Rücksicht auf den

---

[256] Durch die Einführung des Personal Computers war diese erste Forderung der Hacker-Ethik im Bereich der Hardware bereits erfüllt: Der Besitzer eines PCs hat zu diesem bereits vollständigen und unbegrenzten Zugang. Durch die GNU GPL wurde dies nun auch für Software erfüllt; man könnte also statt von „Free Software“ auch von „Personal Software“ sprechen.

[257] STALLMAN: The GNU Project, a.a.O., S. 15. Die Tatsache, daß Stallman in seinem Büro am MIT *wohnte*, mag ihm auch ein anderes als das konventionelle Bild von „Nachbarschaft“ vermittelt haben.

[258] Vgl. etwa STALLMAN: Free Software: Freedom…, a.a.O., S. 164; STALLMAN, Richard Matthew: Why Software Should Be Free. In: Ders., a.a.O., 119-132, hier S. 120; sowie STALLMAN: The GNU Project, a.a.O., S. 16.

[259] Im Englischen wird das Wort „neighbor“ für das biblische „Nächster“ verwendet, etwa bei Matthäus 22,39: „Thou shalt love thy neighbour as thyself“.

[260] STALLMAN: Free Software: Freedom…, a.a.O., S. 164.

ersten Urheber.[261] Stallman sah keinen Sinn darin, solche „freien“ Programme zu entwickeln; seiner Ansicht nach mache sich ein Programmierer so zum „Fußabtreter“[262]: „And then, what have you accomplished? You’ve only made a donation to some proprietary software project.”[263]

Stallman suchte deshalb einen Weg, das Copyright auszuüben, ohne die Freiheit der Nutzer einzuschränken, wie es seiner Meinung nach andere Softwarehersteller taten. Das Copyrightsystem, das zum einen ein Anreiz zu schriftstellerischer Tätigkeit sein sollte, zum anderen durch eine Befristung[264] den allgemeinen Fortschritt sichern sollte, hielt er unter anderem wegen des schnellen Wandels in der Branche für nicht übertragbar auf Software:

> “But in the age of the computer networks, the method is no longer tenable, because it now requires draconian laws that invade everybody’s privacy and terrorize everyone. You know, years in prison for sharing with your neighbor. It wasn’t like that in the age of the printing press. Then copyright was an industrial regulation. It restricted publishers. Now, it’s a restriction imposed by the publishers on the public. So, the power relationship is turned around 180 degrees, even if it’s the same law.”[265]

Mit der GPL drehte er deshalb die Machtverhältnisse um weitere 180 Grad herum: Er übte das Copyright aus, indem er das erlaubte, was üblicherweise durch das Copyright verboten wird: Kopieren, Verändern, Weiterverbreiten – unter einer Bedingung: Alle Kopien und alle Werke, die auf der von der GPL geschützten Software basierten, mußten wiederum nach der gleichen Lizenz und inklusive Quellcode freigegeben werden. So heißt es in der GPL:

> “You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.”[266]

Kritiker sprechen deshalb von der „viralen“ Natur der GPL.[267] STALLMAN selbst zitiert Beispiele von Firmen, die sein Compilerprogramm um neue Schnittstellen erweiterten

---

[261] Genau das war mit dem „X Window System“ geschehen; einer grafischen Benutzeroberfläche, die am MIT entwickelt und freigegeben worden war, in der Hoffnung, sie so zum Standard zu machen. Doch die Hersteller von Unix-Rechnern hatten X nicht mehr freigegeben, nachdem sie es so verändert hatten, daß es auf ihrer Hardware lief; statt dessen hielten sie den Code geheim und verkauften X als eigenes Produkt. Stallman nennt neben dem 1987 freigegebenen X-Window-System das TeX-Layoutprogramm als Beispiel für solcherart korrumpierte freie Software. Vgl. ebd., S. 168f.

[262] „I wanted to give the community a feeling that it was not a doormat, a feeling that it was not prey to any parasite who would wander along.“ Ebd.

[263] Ebd.

[264] In Deutschland greift das Urheberrecht bis zu 70 Jahre nach Tod des Autors.

[265] STALLMAN: Free Software: Freedom…, a.a.O., S. 182.

[266] Abschnitt 2b der GPL. In: STALLMAN: Free Software, Free…, a.a.O., S. 196.

[267] Vgl. WILLIAMS, a.a.O., S. 16. sowie MÖLLER, Erik: Die heimliche Medienrevolution. Wie Weblogs, Wikis und freie Software die Welt verändern. Hannover: heise, 2004, S. 97. Erfunden hat die Bezeichnung „viral“ für die GPL Microsoft-Vizepräsident Craig Mundie. Allerdings wird sie inzwischen sogar von Verfechtern der Open-Source-Idee verwendet. Vgl. RAYMOND: The Art…, a.a.O.

und diese zunächst nicht freigeben wollten – die er aber unter Verweis auf die GPL dazu zwingen konnte.[268]

> "The GNU GPL is not Mr. Nice Guy. It says 'no' to some of the things that people sometimes want to do. There are users who say that this is a bad thing – that the GPL 'excludes' some proprietary software developers who 'need to be brought into the free software community.' But we are not excluding them from our community; they are choosing not to enter."[269]

Die Methode der GNU GPL, abgeleitete Werke unter die gleichen Copyrightbestimmungen zu zwingen, bezeichnet Stallman als „Copyleft", nach einem Begriff, den er einmal auf dem Aufkleber einer Science-Fiction-Messe gelesen hatte.[270] Als Parodie auf die übliche Klausel „Copyright ©, All Rights Reserved" hieß es dort:

Copyleft ↄ, All Rights Reversed

Später wurde das spiegelbildliche ↄ zum Symbol für Stallmans Copyleft-Prinzip.

WILLIAMS sieht diese Regelung in der GPL als einen "Hack" des Urheberrechts an: "Instead of viewing copyright law with suspicion, hackers should view it as yet another system begging to be hacked."[271]

Insgesamt darf jedoch nicht vergessen werden, daß es sich bei den Bedingungen der GPL „nur" um eine Ausübung des Urheberrechts handelt, und nicht etwa um einen Vertrag. Werden diese Bedingungen gebrochen, handelt es sich um eine Urheberrechtsverletzung, die vom Halter des Copyrights beanstandet und eingeklagt werden kann. Der Halter des Copyrights ist dabei derjenige, der das fragliche Programm verfaßt hat, nicht etwa der Autor der GPL oder die Free Software Foundation. Geklagt werden kann auch lediglich auf Unterlassung der unrechtmäßigen öffentlichen Verbreitung; wer das GPL-geschützte Programm für persönliche Zwecke oder nur innerhalb einer Firma verändert hat und dort verwendet, verletzt nicht das Urheberrecht.[272]

2.3.3.3 Die Bedeutung der GPL

Stallmans GPL fand schnell Anklang bei zahlreichen Programmierern außerhalb des GNU-Projekts. Neben ihrer Fairneß überzeugte vor allem ihre juristische Versiertheit.[273] Wie aufgrund der Bedeutung von Unix vorherzusehen war, weckte das Projekt auch das

---

[268] Es waren dies die Firma Xerox und das amerikanische Industriekonsortium MCC, die Stallmans GCC Compiler um ein C++-bzw. Objective-C-Frontend erweiterten. Vgl. STALLMAN, Richard Matthew: Copyleft: Pragmatic Idealism. In Ders.: Free Software, Free Society. Selected Essays. Boston: GNU-Press, 2002, S. 91-94, hier S. 91.

[269] Ebd., S. 92.

[270] Den Sticker hatte ihm ein anderer Programmierer, Don Hopkins, zugeschickt. Vgl. WILLIAMS, a.a.O., S. 128.

[271] Ebd.,. 127.

[272] Vgl. WIKIPEDIA (engl.): GPL. In: http://en.wikipedia.org/wiki/GPL [Aufgerufen am 8.12.2004].

[273] „A few years after the release of the GPL, Perens says, he decided to discard Electric Fence's homegrown license in favor of Stallman's lawyer-vetted copyright. 'It was actually pretty easy to do,' Perens recalls." WILLIAMS, a.a.O., S. 130.

Interesse von Hochschulen: Angeregt von GNU[274] entschied die Universität von Berkeley im Juni 1989, ihre Unix-Software unter besonderen Bedingungen (BSD-Lizenz[275]) freizugeben – noch heute finden sich deshalb viele BSD-Programme in GNU/Linux.

Tim Berners-Lee, der Erfinder des World-Wide-Webs, war von der GNU GPL so überzeugt, daß er 1992 sogar die Seitenbeschreibungssprache HTML durch diese Lizenz schützen lassen wollte:

> "During the preceding year I had been trying to get CERN to release the intellectual property rights to the Web code under the General Public License (GPL) so that others could use it."[276]

Allerdings weigerte sich damals die Firma IBM die Sprache HTML zu unterstützen, wenn es irgendeine lizenzrechtliche Klausel gäbe;[277] Berners-Lee entschied sich deshalb, HTML als „public domain" freizugeben.[278]

In Deutschland ist die Wirksamkeit der GNU General Public License (GPL) seit Mitte 2004 juristisch teilweise anerkannt – nach einem Urteil des Landgerichts München gegen den Netzwerkausrüster Sitecom.[279] In der Urteilsbegründung heißt es:

> „Die Kammer teilt zunächst die Auffassung, daß in den Bedingungen GPL (General Public Licence) keinesfalls ein Verzicht auf Urheberrechte und urheberrechtliche Rechtspositionen gesehen werden kann. Im Gegenteil bedienen sich die Nutzer der Bedingungen des Urheberrechts, um ihre Vorstellungen der weiteren Entwicklung und Verbreitung von Software sicherzustellen und zu verwirklichen."[280]

Diesem Urteil zufolge greift Stallmans Copyleft-Prinzip in Deutschland vollständig: Wer seine Änderungen nicht wiederum unter die GPL stellt und sie nicht im Quellcode veröf-

---

[274] WILLIAMS zitiert dazu den BSD-Programmierer und Angestellten der Universität von Berkeley, Kalifornien mit den Worten: „'I think it's highly unlikely that we ever would have gone as strongly as we did without the GNU influence,' says Bostic, looking back. „It was clearly something where they were pushing hard and we liked the idea.'" Ebd.

[275] Die BSD-Lizenz verzichtete auf die Copyleft-Methode und enthielt nur die Bedingung, die Universität zu nennen, falls für das aus ihrem Code entstandene Produkt Werbung gemacht würde. 1999 entschied sich die Universität allerdings, diese Klausel fallen zu lassen: Aufgrund der kaum zu überblickenden Vielzahl der durch diese und ähnliche Lizenzen anderer Universitäten freigegebenen Programme konnte die Bedingung bei Werbeaktionen nicht mehr befolgt werden, was bedeutete, daß diese wenig überdachte Lizenzklausel ohnehin permanent gebrochen wurde. Ebd., S. 140.

[276] BERNERS-LEE, Tim: Weaving the Web. The Past, Present and Future of the World Wide Web by its Inventor. London, New York: Texere, 2000, S. 80.

[277] Quae nocent, docent.

[278] BERNERS-LEE, a.a.O., S. 80.

[279] Urteil Landgericht München I vom 19.05.2004 (Az. 21 O 6123/03). Der Hersteller hatte in seinen Netzwerkgeräten Software des Open-Source-Projects „netfilter/iptables" installiert, sie beinhalten grundlegende Weiterleitungs- und Sicherheitsfunktionen für den Netzwerkverkehr, ohne die die Geräte nicht funktionsfähig wären. Nach der GPL hätte Sitecom den Quellcode der Software mit dem Gerät ausliefern müssen, also auch alle herstellerspezifischen Anpassungen, dies tat er nicht. Vgl.: c't Magazin für Computertechnik: Verfügung wegen GPL-Verletzung. Ausgabe 10/2004. Hannover: heise, 03.05.2004, S. 44. und Linux Magazin: Netfilter-Projekt verteidigt GPL. Ausgabe 06/04. München: Linux New Media AG, 06.05.2004, S. 21.

[280] Aus dem Urteil des Landgerichts München 1 vom 19.05.2004 (Az. 21 O 6123/03). Zitiert nach der schriftlichen Urteilsbegründung, veröffentlicht durch die Anwaltskanzlei JBB Berlin/München unter: http://www.jbb.de/urteil_lg_muenchen_gpl.pdf
[Aufgerufen am 06.08.2004], S.12. Im weiteren Verlauf der Urteilsbegründung heißt es außerdem: „Das Grundprinzip der Open Source Software wird überdies von dem Gesetzgeber mit der Regelung in § 32 Abs. 3 S. 3 UrhG ausdrücklich anerkannt". Ebd., S. 19.

fentlicht, verliert jedes Nutzungsrecht an der betreffenden Software. Das Gericht sieht darin auch keine Härte, da „der Verletzer die Rechte durch Annahme und Befolgung der Bedingungen jederzeit wieder erwerben kann. Der automatische Verlust ist daher für den Verletzer nicht besonders gravierend.“[281]

Die GPL schützt heute fast 70 Prozent aller freien Softwareprojekte.[282]

2.3.4 Die Free Software Foundation

Um das GNU-Projekt voranzubringen, hatte Stallman 1985 nach Fertigstellung von GNU Emacs die *Free Software Foundation* (FSF) gegründet. Diese steuerbefreite Non-Profit-Organisation übernahm den Verkauf von GNU Emacs und weiteren GNU-Programmen sowie den dazugehörigen Handbüchern.[283] Aus dem Erlös stellte sie zunächst Programmierer an, um die teilweise undankbaren[284] Aufgaben, die mit der Programmierung des neuen Betriebssystems einhergingen, zu erfüllen. Heute beschäftigt sich die FSF vor allem mit der Überwachung ihrer Urheberrechte, der Durchsetzung der GPL für die eigenen GNU-Programme sowie der Weiterentwicklung der GPL.[285]

---

[281] Ebd., S. 18. Unklar ist dagegen, ob auch die Haftungs- und Gewährleistungsklauseln gemäß den Ziffern 11 und 12 der GPL gültig sind. Sie schließen Haftung und Gewährleistung selbst bei Vorsatz und grober Fahrlässigkeit aus, was nach Ansicht von Experten nach europäischem Recht ausgeschlossen ist, und selbst dann nicht möglich wäre, wenn man der Nutzung der (kostenlosen) Software das Schenkungsrecht zugrunde legen würde. KNOP verweist in diesem Zusammenhang auf ein entsprechendes Gutachten des Göttinger Juraprofessors Gerald Spindler im Auftrag des Verbands der Software-Industrie Deutschland. Vgl. KNOP, Carsten: Ein Juraprofessor schreckt die Linux-Welt auf. Streit um Rechtsunsicherheiten von Open-Source-Software etwa von Suse oder MySQL. Frankfurter Allgemeine Zeitung, 14.08.03, S. 16 (Wirtschaft). In Frankreich wird deshalb derzeit eine entsprechend abgeänderte Version erarbeitet. Maßgeblich beteiligt an dieser neuen Lizenz namens CeCILL sind öffentlich finanzierte Institutionen: Das Kommissariat für Atomenergie (CEA), das nationale Zentrum für naturwissenschaftliche Forschung (CNRS) und das Forschungszentrum für Informatik und Automatisierung (INRIA). Die Free Software Foundation wurde nicht einbezogen. Vgl. LINUX Magazin: Frankreich will neue Open-Source-Lizenz. Ausgabe 09/04. München: Linux New Media AG, 05.08.2004, S. 16.

[282] Vgl. HEINZE, Daniel; KELLER, Alexander: Der Preis der Freiheit – was Softwareentwickler über Open-Source-Lizenzen wissen sollten. In: SAUERBURGER, Heinz (Hrsg.): Open-Source-Software. Heidelberg: dpunkt, 2004, S. 47. HEINZE / KELLER zitieren dazu eine Statistik des Software-Archivs „freshmeat“, bei dem am 11. Juli 2004 68,8 Prozent der 35.945 Open-Source-Softwarepakete durch die GPL geschützt waren. Bei dem zweiten bedeutenden Software-Archiv, „SourceForge“ unterlagen im Herbst 2004 MÖLLER zufolge 67,62 Prozent der 92.181 Projekte der GPL. Vgl.: MÖLLER, a.a.O., S. 95.

[283] Wie bereits erwähnt, ging es um den Verkauf der physischen Datenträger (Bänder). Die freie Software selbst konnte dann beliebig kopiert werden Zu diesem Zeitpunkt hatten hauptsächlich Hochschulen, Großunternehmen und Militär Zugang zum ARPA- bzw. Internet und die Bandbreite war gering, so daß es eine Nachfrage für dieses Angebot gab. Vgl. STALLMAN, Richard Matthew: The GNU Project, a.a.O., S. 21.

[284] Das Unix-Betriebssystem besteht aus vielen kleinen Einzelkomponenten, die teilweise triviale Aufgaben erfüllen. Es ist also nicht unbedingt eine professionelle Herausforderung, diese Komponenten nachzubilden; allerdings gab es keine Alternative, wie Stallman in einer Rede vor (amüsiertem) Publikum erklärt: „[...] we knew that we wouldn't have a complete system without it. And even if it was totally boring and unromantic, like tar or mv, (audience laughs) we did it. Or ld – you know there's nothing very exciting in ld, but I wrote one. (audience laughs) And I did make efforts to have it do a minimal amount of disk I/O so that it would be faster and handle bigger programs. But, you know, I like to do a good job; I like to improve various things about the program while I'm doing it. But the reason that I did it, wasn't that I had brilliant ideas for a better ld. The reason I did it is that we needed one that was free.“ STALLMAN: Free Software: Freedom…, a.a.O., S. 21.

[285] Die erste Version der GPL erschien 1989, die zweite 1991, die dritte ist zum Zeitpunkt der Niederschrift dieser Arbeit in Arbeit.

Auffallend bleibt vor allem die organisatorische Zielstrebigkeit, mit der Stallman sein GNU-Projekt anging; dies widerspricht Beobachtungen,[286] wonach die meisten freien Projekte spontan begonnen werden, um gerade anstehende Probleme ihrer Urheber zu lösen. Dagegen stellt Stallman klar: "[...] many essential pieces of GNU software were developed in order to have a complete free operating system. They come from a vision and a plan, not from impulse."[287]

Aus dem Erlebnis des unsozialen Verhaltens, das die Einführung proprietärer Software am MIT ausgelöst hatte, definiert Stallman freie Software als soziale Bewegung und grenzt es dadurch von den Verfechtern des Begriffs „Open Source" ab:[288]

> "As one person put it, 'Open Source is a development methodology; free software is a social movement.' 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."[289]

Die Notwendigkeit freier Software begründet Stallman mit den Grundregeln sozialen Zusammenlebens sowie mit Kants Goldener Regel[290], dem Gebot der Nächstenliebe[291], dem jüdischen Religionsführer Hillel dem Älteren[292] und dem tschechischen Freiheitskämpfer Václav Havel[293].

Für sein Engagement wurde Stallman 1990 mit dem MacArthur Fellowship-Preis in Höhe von 240.000 Dollar ausgezeichnet.[294] Mit der neugewonnen finanziellen Freiheit beschloß er, weltweit für das GNU-Projekt zu werben. Ohne es zu wissen, führte dieser Entschluß zum größten Erfolg für das Projekt, aber auch zum Beginn tragischer Konkurrenz-

---

[286] In seinem Essay „The Cathedral and the Bazaar" behauptet RAYMOND in einer seiner Grundregeln für Open Source: „Every good work of software starts by scratching a developer's personal itch". RAYMOND: The Cathedral…, a.a.O., S. 23.

[287] STALLMAN: The GNU Project, a.a.O., S. 24.

[288] Vgl. Kapitel 3.5

[289] STALLMAN, Richard Matthew: Why „Free Software" is Better than „Open Source". 1998. In: Ders.: Free Software, Free…, a.a.O., S. 55-60, hier S. 55.

[290] „The reason a good citizen does not use such destructive means to become wealthier is that, if everyone did so, we would all become poorer from the mutual destructive-ness. This is Kantian ethics; or, the Golden Rule. Since I do not like the consequences that result if everyone hoards information, I am required to consider it wrong for one to do so. Specifically, the desire to be rewarded for one's creativity does not justify depriving the world in general of all or part of that creativity." STALLMAN: The GNU Manifesto, a.a.O., S. 37.

[291] STALLMAN: Free Software: Freedom…, a.a.O., S. 164.

[292] So zitiert er als Motiv für das GNU-Projekt aus Hillels Pirkei Avot (Sprüche der Väter) 1:14: „If I am not for myself, then who will be for me? And if I am only for myself, then what am I? And if not now, when?" STALLMAN: The GNU Project, a.a.O., S. 164.

[293] „Vaclav Havel has advised us to 'Work for something because it is good, not just because it stands a chance to succeed.' A business making proprietary software stands a chance of success in its own narrow terms, but it is not what is good for society." STALLMAN, Richard Matthew: Why Software…, a.a.O., S. 127.

[294] Außer ihm erhielt den Preis im gleichen Jahr unter anderem die Autorin Susan Sontag.

kämpfe: Einen seiner Vorträge hielt Stallman 1991[295] an der Polytechnischen Universität in Helsinki, Finnland. Unter den Anwesenden: Ein 21jähriger Student namens Linus Torvalds.

2.4 Zusammenfassung

Die wachsende Verbreitung von Computern und der sich Ende der 70er Jahre abzeichnende Markterfolg von Software führten dazu, daß Programme als Wirtschaftsgüter verstanden wurden. Die daraus resultierenden Folgen wie Geheimhaltungspflichten und Nutzungseinschränkungen waren mit dem bis dahin vorherrschenden, enthusiastischen Selbstverständnis der Berufsgruppe der Programmierer, d.h. der Hacker-Ethik, nicht vereinbar. Am AI Lab des MIT führte diese Entwicklung dazu, daß durch den Versuch, ein wissenschaftliches Projekt zu verwirtschaftlichen, eine Wissenskultur gespalten und zerstört wurde, die zuvor jahrzehntelang hochinnovative Entwicklungen vorangetrieben hatte.

Als direkte Reaktion darauf begann 1983 mit dem GNU-Projekt der Versuch, die Einhaltung der Werte der Hacker-Ethik durch eine neue Form des Copyrights, der „GNU General Public License“ (GPL), zu erzwingen; Grundüberzeugungen der Hacker flossen in die Definition Freier Software ein.[296] Als Produkt wurde Unix ausgewählt, was sich wegen der wachsenden Bedeutung dieses Betriebssystems für Unternehmen und Hochschulen, aber auch für das spätere Internet als geschickter Schachzug erwies.

Die GPL sorgte zum einen dafür, daß mit ihr geschützte Software nicht mehr „unfrei“ gemacht werden konnte, zum anderen ließ sie bewußt Spielraum für eine kommerzielle Nutzung. Diese beiden Unterschiede zu anderen „hausgemachten“ Copyright-Vermerken ihrer Zeit waren verantwortlich für ihre weite Verbreitung. Richard M. Stallman, der Gründer dieses Projekts, sah die Verbreitung freier Software aber auch als wichtige gesellschaftliche Aufgabe an und definierte sich und seine Sympathisanten als soziale Bewegung. Vor allem das unterscheidet ihn von Linus Torvalds, dem Erfinder von Linux und den Anhängern des Begriffs „Open Source“.

---

[295] Es existieren widersprüchliche Angaben, ob dieser Vortrag 1990 oder 1991 gehalten wurde. So schreibt WILLIAMS zwar an einer Stelle: „In 1990, Stallman paid a visit to the Polytechnic University in Helsinki, Finland.“ (WILLIAMS, a.a.O., S. 136); drei Seiten weiter heißt es dagegen, Torvalds habe die Rede 1991 gehört (Ebd., S. 139). Torvalds zufolge fand der Vortrag 1991 statt (Vgl. TORVALDS, Linus Benedict: Just for fun. Wie ein Freak die Computerwelt revolutionierte. München: Hanser, 1991, S. 66); dies wird unterstützt durch andere Berichte über eine Europareise Stallmans 1991, so schreibt BRAUER über einen Vortrag Stallmans mit dem Titel „New Monopolies on Writing Programs – How to Protect Your Freedom to Write Software“ vom 25.06.91 in München. BRAUER, Wilfried: Tätigkeitsbericht 1991. Theoretische Informatik und Grundlagen der KI. München: Institut für Informatik, Technische Universität München, 1991. Veröffentlicht online unter: http://wind.in.tum.de/taetigkeitsberichte/taet91/taet91.html [Aufgerufen am 08.01.2005].
Unter dem Pseudonym PI hat das Mailbox.Magazin Chalisti einen Bericht über einen Vortrag Stallmans ebenfalls am 25.06.91 in Frankfurt a. M. veröffentlich. PI (pi@helpdesk.rus.uni-stuttgart.de): RMS ueber User Interface Copyrights und Software Patente. In: Chalisti, Onlinemagazin, Ausgabe 15, 24.7.1991. Veröffentlicht unter: http://www.chscene.ch/ccc/chalisti/chalisti_15/008_OFA8.html [Aufgerufen am 08.01.2005].

[296] Es ist nicht auszuschließen, daß LEVY's 1984 erschienenes Buch „Hackers“, das die erste ausführliche Darstellung der Hacker-Ethik bildet, eine Rolle bei der Entscheidung Stallmans für das GNU-Projekt und seiner Definition freier Software gespielt hat.

3 GNU/Linux und Open Source

Der Begriff „Open Source“ ist eine Anspielung auf die Tatsache, daß Computerprogramme nicht in der gleichen Sprache geschrieben werden, in der Computer sie ausführen. Während das Vorprodukt, der in einer Programmiersprache verfaßte Quellcode (engl. „Source“), für Programmierer gut verständlich ist, ist das Endprodukt, die Binärdatei, die der Prozessor zu sehen bekommt, unleserlich.[297]

Mit dem Prozeß des „Kompilierens“, des Umwandeln des Quellcodes in eine Binärdatei, geht also auch eine *Entwertung* einher: Binärdateien sind notwendig für den Endanwender; für Programmierer, die die Software studieren oder ändern wollen, sind sie unbrauchbar. Das Offenlegen des Quellcodes ist also zunächst und vor allem im Interesse von Programmierern, genauer gesagt: von fremden Programmierern.

Für denjenigen, der die Software selbst geschrieben hat, für die Herstellerfirma, sowie für den reinen Benutzer hat das Offenlegen des Quellcodes dagegen *unmittelbar* keinen Vorteil: Der Programmierer gibt eine geistige Leistung, die Zeit, Mühe und eine Berufsausbildung gekostet hat, kostenlos an fremde Programmierer weiter, die diese per Mausklick in Sekundenbruchteilen kopieren können. Der Hersteller verliert neben seiner Investition die exklusive Vertriebsmöglichkeit für die Software, bzw. einen Wettbewerbsvorteil, falls es sich bei dem Programm um eine firmeninterne Lösung handelt. Für den Anwender ist ein Nutzen ebenfalls nicht sofort ersichtlich: Zwar ist freie Software häufig kostenlos oder günstiger als proprietäre Software, das spielt aber nur eine Rolle, falls der Anwender noch keine Produktentscheidung getroffen hat.[298] Im Falle eines Umstiegs verliert er seine bisherige Investition,[299] vor allem aber einen Großteil seiner mit dem proprietären Produkt gewonnenen Erfahrung.

Es ist der Verdienst der modernen Verfechter von freier Software, die sich inzwischen vor allem unter dem Begriff „Open Source“ zu Wort melden, die erheblichen *mittelbaren* Vorteile deutlich gemacht zu haben, die sowohl für Programmierer, Hersteller und Nutzer aus einer Freigabe des Quellcodes entstehen können, als auch für die Gesellschaft, in der solche Software genutzt wird. Genannt werden unter anderem eine höhere Qualität und Sicherheit, die Vermeidung der Abhängigkeit von einem Softwarelieferanten, ein größerer

---

[297] Für eine genaue Erklärung dieses Vorgangs siehe den Anhang: „Das Schwein guckt ins Uhrwerk oder: Was ist Quellcode?"

[298] Dabei kommt es darauf an, welchen Zahlungsbedingungen er unterliegt. Hat er die Softwarelizenz gegen Einmalzahlung erworben, würde er mit einem Umstieg diese Investition verlieren. Muß er jährlich zahlen, entweder aufgrund eines Abonnements oder aufgrund von notwendigen Updates, kann sich der Umstieg auf ein günstigeres Produkt lohnen. Allerdings müssen dabei auch die Kosten für den Umstieg selbst eingerechnet werden.

[299] Wenn er überhaupt eine getätigt hat: Nach einer Studie des Marktforschungsinstituts IDC waren 2003 in Deutschland 30 Prozent aller eingesetzten Standardprogramme raubkopiert; in Frankreich, Italien oder Spanien sogar mehr als 40 Prozent. Zitiert nach: Business Software Alliance: Studie: IDC Pirateriezahlen. Veröffentlicht unter:
http://www.bsa.org/germany/piraterie/piraterie.cfm [Aufgerufen am 11.12.2004].
In einem Artikel über die Fertigstellung der Open-Source-Software „OpenOffice“ in einer Version in Suaheli heißt es, es sei schwer, in Tansania die Vorteile freier Software zu erklären: „‘People here don't buy Microsoft licences, so free software is a difficult concept to explain as they think Microsoft is also free,’ said Alberto Pascual [der Leiter des Suahili-Lokalisierungs-Projektes ‘Kilinux'].“ MARSON, Ingrid: OpenOffice.org goes Swahili. ZDNet UK, December 06, 2004, 16:05 GMT. Veröffentlicht unter:
http://news.zdnet.co.uk/software/linuxunix/0,39020390,39179058,00.htm [Aufgerufen am 11.12.2004]. Es ist dabei anzumerken, daß es noch nie ein Microsoft-Produkt auf Suaheli gegeben hat.

Marktanteil, das Setzen von offenen Standards (und damit Investitionssicherheit), eine enge Zusammenarbeit zwischen Entwicklern und Anwendern sowie eine bessere gesellschaftliche Kontrolle.[300] Im Unterschied zu den Forderungen Stallmans stützt sich die Argumentation nicht mehr auf einen Appell an das Sozialverhalten von Programmierern untereinander oder auf die technische Utopie der Ubiquität von Software („like air") sondern bewußt auf unternehmerische, organisationstheoretische und volkswirtschaftliche Bewertungen.[301]

Der Name „Open Source" drückt explizit noch keinen Unterschied zur Free-Software-Bewegung aus, denn auch dieser ging es um eine Offenlegung des Quellcodes. Eine implizite Bedeutung steckt im Fehlen des Wortes „Frei". Die radikalste Veränderung aber bestand darin, daß überhaupt ein neuer Begriff gesucht und gefunden wurde: Denn das geschah ausdrücklich als Alternative zu „freier Software" und mit dem Ziel, Richard M. Stallman die Deutungshoheit für diese Art Software zu entreißen.

Im folgenden wird soll nicht nur die Entstehung von GNU/Linux und Open Source beschrieben werden, sondern auch die Veränderungen im Verständnis von freier Software und ihrer Entwicklung.

3.1 Das Versagen der "Unix-Hacker"

Es ist möglicherweise berechtigt, den Journalisten LEVY, WILLIAMS und MOODY,[302] die sich in Buchform mit der Geschichte der alten und neuen Hacker auseinandergesetzt haben (und auch manchen Wissenschaftlern[303]), eine gewisse Verkürzung vorzuwerfen, was ihre Darstellung der Rolle Stallmans angeht: Bei der Berichterstattung über einen Besuch Stallmans in Frankfurt habe ich selbst erfahren, wie schwer es fällt, Stereotypen zu vermeiden; etwa die des „letzten Hackers", des „Propheten freier Software" oder schlicht des „Gurus" – vor allem, wenn diese quasi-religiösen oder mythischen Bilder bereits zuvor in Artikeln und Texten verbreitet wurden und noch mehr, wenn dies der Grund ist,

---

[300] Vgl. RAYMOND, Eric Steven: The Magic Cauldron. In: The Cathedral and the Bazaar. Musings on Linux and Open Source by an Accidental Revolutionary. Sebastopol: O'Reilly, 2001, S. 134-140 und EBERT, Christof; RUFFIN, Michel: Produkte entwickeln mit Open-Source-Software – Risiken und Erfahrungen. In: SAUERBURGER (Hrsg.), a.a.O., S. 30f.

[301] Der GNU/Linux-Aktivist und wichtigste Fürsprecher des Begriffs „Open Source", Raymond, richtete seine Pressearbeit gezielt auf Finanz- und Wirtschaftsmedien: „It was during this time [Juli bis November 1998] that we started to see fairly steady coverage from the financial media I had originally targeted, led off by articles in the Economist and a cover story in Forbes." RAYMOND, Eric Steven: Revenge of the Hackers. In: Ders.: The Cathedral…, S. 167-192, hier S. 183.

[302] LEVY, a.a.O.; Williams, a.a.O.; Moody, a.a.O.

[303] Vgl. solcherart verkürzte Darstellungen bei: GRASSMUCK, a.a.O., S. 218-226. REITER, Bernhard E.: Wandel der IT. Mehr als 20 Jahre freier Software. In: SAUERBURGER, Heinz (Hrsg.): Open-Source-Software. Heidelberg: dpunkt, 2004, S. 83-110, hier S. 84. MERETZ, Stefan: LINUX & CO. Freie Software – Ideen für eine andere Gesellschaft. Neu-Ulm: AG SPAK, 2000. Online-Veröffentlichung auf: http://www.kritische-informatik.de/index.htm?fsrevol.htm [Aufgerufen am 13.12.2004].

warum man die Story überhaupt verkaufen konnte.[304] Stallmans Charisma, seine Konsequenz, aber auch seine bis zu ungenierter Unhöflichkeit[305] reichende Schroffheit sowie die offensichtliche Ehrfurcht seiner Anhänger bieten für solche Stereotypen die willkommene Rechtfertigung.

Raymond, der mit Stallman während der ersten Jahre des GNU-Projekts eng zusammenarbeitete, kritisiert solche Darstellungen, „[...] especially the tendency to portray Stallman as the sole link between what came before him and the post-1985 hacker culture".[306] Er sieht sie als Folge einer Selbstinszenierung Stallmans:[307]

> "I was a hacker at the time, and I knew many others. We were not at all affected by the "Symbolics War" that so scarred Richard because the ground we were standing on was different – we were Unix guys. [...] The MIT group was important, but they were not the only hackers in the world. [... Stallman] tends to downplay the cultural context in which he operated – one consisting not merely of other MIT hackers [...] but of hackers in other traditions – the Unix guys, the Xerox PARC crowd, the SAIL group. [...] Most of the rest of us old-timers have given up trying to remind the world that the hacker culture did *not* in fact collapse in 1980-1983 only to spring from RMS's brow reborn with the GNU manifesto; I sometimes feel like a last holdout myself on this one :-)." [308]

In der Tat wäre es wenig einleuchtend, den heutigen Erfolg freier Softwareprojekte, der der Beteiligung einer Vielzahl von Programmierern zu verdanken ist, ausschließlich auf

---

[304] Es ist schwer, sich der Pseudoreligiösität und -mythologie zu entziehen, die in den Darstellungen von Stallmans Rolle mitschwingt und die man übertrieben folgendermaßen darstellen könnte: Vertrieben aus dem Paradies der ursprünglichen Hacker-Community wird Stallman zunächst zum Letzten seiner Art und dann zum Rufer in der Wüste. Der Zufallsfund des Linux-Kernels durch den hedonistischen „idiot savant" Linus Torvalds fügt zu Stallmans einsamen Leistungen das letzte fehlende Puzzlesteinchen hinzu und die Hacker-Kultur erhebt sich wie Phönix aus der Asche. Wie es aber eben mit Religionsführern so geht, wird „St. IGNUcius" (selbstgewählter Spitzname Stallmans) im Augenblick des Triumphs von der „Open-Source-Bewegung" verraten, wobei deren Sprecher, sein früherer Freund Eric Steven Raymond, die Rolle des „Judas" einnimmt (wenn auch eher die des reflektierten Judas aus „Jesus Christ Superstar").

[305] Wie geht man mit jemandem um, der als Stargast einer Podiumsdiskussion während des gesamten Verlaufs dieser Diskussion mitten auf dem Podium in seinen Laptop starrt, tippt und Emails beantwortet und nur gelegentlich auf Stichworte reagiert? Vgl. HANSEN, a.a.O.

[306] RAYMOND, Eric Steven: A Fan of Freedom: Thoughts on the Biography of RMS. 11.11.2003. Online veröffentlicht unter:
http://www.catb.org/~esr/writings/rms-bio.html [Aufgerufen am 13.12.2004].

[307] RAYMOND vermißt dabei ihm selbst vertraute Aspekte Stallmans, etwa dessen Engagement als Science-Fiction-Fan oder als „fairly ordinary-looking young man with short(!) hair who between 1977 and 1980 or so chased several of my girlfriends and (despite RMS's later report of his consistent lack of success with women) caught at least two of them". RAYMOND: A Fan…, a.a.O.
Der Vorwurf der Selbstinszenierung ist unter dem Vorbehalt zu bewerten, daß RAYMOND an gleicher Stelle einräumt, sich ebenso zu inszenieren. Auch überzeichnet er in seinen Schriften Stallman als „blinden Eiferer" (engl. „zealot") und benutzt ihn als Kontrastmittel, um Open Source als undogmatisch darzustellen. Vgl. RAYMOND, Eric Steven: Homesteading the Noosphere. In: Ders.: The Cathedral…, a.a.O., S. 65-111, hier S. 68f.

[308] RAYMOND: A Fan…, a.a.O.

die Arbeit eines Einzelnen zurückzuführen, zumal wenn dessen Führungsqualitäten strittig sind.[309]

Statt dessen ist festzustellen, daß es beim Start des GNU-Projekts 1983 sehr wohl noch eine weltweite Hacker-Kultur gab. Viele dieser Programmierer beschäftigten sich mit Unix. Grundsätzlich begrüßten sie die Idee, ein derartiges System als freie Software nachzubilden, allerdings reagierten sie zurückhaltend auf den Urheber:[310] Stallman war ITS-Hacker, kein Unix-Hacker und das Ende seiner Community am MIT betraf die Unix-Programmierer kaum; Unix-Versionen von AT&T, der Berkeley-Universität und der neuen, 1982 gegründeten Firma Sun versprachen ihnen im Gegenteil vielfältige Entwicklungsmöglichkeiten.

Anders als Stallman übersahen die Unix-Hacker, daß die beschriebenen Kommerzialisierungsprozesse begannen, auch ihre Welt zu beeinträchtigen: Die Zerschlagung von Bell 1983 ermöglichte es AT&T, Unix als Produkt zu vermarkten; dies wiederum führte zu einem langwierigen, auch juristisch ausgefochtenen Konkurrenzkampf zwischen AT&T, Sun und der Universität von Berkeley (erster Unix-Krieg) mit einer Konsequenz für Unix-Hacker: Der Austausch des Quellcodes kam zum Erliegen.

> "What none of us realized at the time was that the productization of Unix would destroy the free exchanges of source code that had nurtured so much of the system's early vitality. Knowing no other model than secrecy for collecting profits from software [...], AT&T clamped down hard on source-code distribution. Bootleg Unix tapes became far less interesting in the knowledge that the threat of lawsuit might come with them. Contributions from universities began to dry up."[311]

Die Einführung des IBM-PCs 1981 berührte die Unix-Welt zunächst kaum: die großen Unix-Hersteller betrachteten die 8086- und 80286-Prozessoren als zu leistungsschwach; eine Meinung, der sich viele Programmierer anschlossen.[312] Microsoft erkannte zwar die Marktchancen für ein Unix auf IBM-PCs – nachdem auf diesen Rechnern aber ihr hauseigenes System MS-DOS Erfolg hatte, beendete die Firma ihr Unix-Engagement.[313]

---

[309] So hatte Stallman am 17.11.2003 Aufsehen erregt, als er einen wichtigen Verantwortlichen des GNU-Projekts, Thomas Bushnell, von seinen Ämtern entband, nachdem dieser Aspekte der „GNU Free Documentation License" (unter der beispielsweise alle Texte in Wikipedia stehen) als zu wenig frei kritisiert hatte. Bushnell warf Stallman daraufhin vor, grundsätzlich die freie Meinungsäußerung im GNU-Projekt zu unterdrücken. Vgl. BUSHNELL, Thomas: What's up with the GFDL? E-Mail vom 17.11.2003, 11:33:16 Uhr, an die Mailingliste „gnu-prog-discuss@gnu.org". Archiviert unter: http://lists.softwarelibero.it/pipermail/discussioni/2003-November/008465.html [Aufgerufen am 12.12.2004].

[310] WILLIAMS: a.a.O., S.103.

[311] RAYMOND: The Art…, a.a.O.

[312] Vgl. ebd. sowie MOODY, a.a.O., S. 63. MOODY zitiert Thomas Roell, den Gründer des X386-Projekts (zur Ansteuerung einer grafischen Benutzeroberfläche) mit den sogar noch auf das Jahr 1990 bezogenen Worten: „Everybody said the Intel-Unix PCs would be just toys".

[313] Die Firma von Bill Gates hatte bereits 1979 (also zwei Jahre vor ihrem Lizenzvertrag mit IBM über MS-DOS) eine Unix-Lizenz von AT&T erworben (AT&T stand damals als Noch-Monopolist unter Kontraktierungszwang). Unter dem Namen „Xenix" vertrieb Microsoft ab Ende 1980 dieses System für 16-bit-Microprozessoren, d.h. auch für 8086-PCs. Damit gehörte Microsoft zu den ersten, die die (aus heutiger Sicht offensichtlichen) Marktchancen von Unix auf billigen PCs erkannten. Nach dem Erfolg von MS-DOS (unter eigener Lizenz) und der Vereinbarung einer weiteren Zusammenarbeit mit IBM (über OS/2) beendete Microsoft 1983 zunächst seinen Abstecher in die Unix-Welt.

Abbildung 17: Der erste IBM-PC 1981.[314]

Als dann ab 1986 der PC seinen Durchbruch am Markt erzielte,[315] mit einem Prozessor (einem „386er"), der zum Ausführen von Unix geeignet war, gab es keins dafür: AT&T und Sun, die beiden großen Unix-Firmen, betrachten PCs nun nicht als Markt, sondern als Bedrohung für ihr Geschäft mit teuren Hochleistungsrechnern. Um sich den Unix-Standard zu sichern, verbündeten sie sich und lieferten sich neue rechtliche und wirtschaftliche Kämpfe mit anderen Anbietern wie IBM, DEC und Hewlett-Packard[316] (zweiter Unix-Krieg) – und schufen damit die Marktlücke für Windows NT.[317]

Die Programmierer weltweit aber, für die Unix in Universitäten und Großunternehmen zum selbstverständlichen Betriebssystem geworden war, mußten ohnmächtig zusehen, wie die Systeme von Microsoft in mittlere und kleine Unternehmen einzogen, und sogar in ihre eigenen Heime.

> "Hardware was getting cheaper, but Unix was still too expensive. We were belatedly becoming aware that the old monopoly of IBM had yielded to a newer monopoly of Microsoft, and Microsoft's mal-engineered software was rising around us like a tide of sewage."[318]

---

[314] Quelle: International Business Machines Corporation. Veröffentlicht online unter: http://www-03.ibm.com/ibm/history/exhibits/pc/pc_1.html [Aufgerufen am 10.10.2005].

[315] 1986 hatte IBM aufgrund von technischen Fehlentscheidungen seine Vormachtstellung auf dem PC-Markt zugunsten der Hersteller von sogenannten „PC-kompatiblen" Rechnern wie Compaq verloren; damit ging ein deutlicher Preisrutsch einher.

[316] Zusammengeschlossen in der „Open Software Foundation" (ein Euphemismus).

[317] Microsoft gelang es damit, in den Unix-Markt vorzustoßen, ohne sich den tobenden Lizenzstreitigkeiten auszusetzen. Das 1988 entwickelte Betriebssystem Windows NT war auf der Basis eines langjährigen technischen Rivalen von Unix entstanden, dem Betriebssystem VMS. Vgl. MOODY, a.a.O., S. 5f sowie WIKIPEDIA (engl.): Unix wars. In: http://en.wikipedia.org/wiki/UNIX_wars [Aufgerufen am 13.12.2004].

[318] RAYMOND: The Art…, a.a.O.

MS-DOS eignete sich nicht zum Hacken.[319] Und für eine Portierung von Unix auf den PC verfügten die Hacker nicht mehr über bezahlbare Rechte am Quellcode. Mit einem Mal war eine Nachfrage nach einem freien Unix-System entstanden – aber es gab keine überzeugenden Angebote:

- Zwar hatte Stallman im Jahr 1988 Mitarbeiter der Universität von Berkeley überzeugen können, ihre eigene Unix-Version ***BSD*** von urheberrechtlich geschütztem Quellcode der Firma AT&T zu befreien. Doch das Entfernen dieser Teile war ein langwieriger Prozeß und hinterließ ein unvollständiges System sowie juristische Probleme. Dennoch begann William Jolitz auf dieser Basis 1990 mit der Anpassung von BSD an den PC. Allerdings mußte dieses „386BSD" genannte Projekt Ende 1991 einen herben Rückschlag einstecken: Die Geldgeber[320] waren mit Jolitz' Absicht nicht einverstanden, 386BSD freizugeben. Jolitz zerstörte daraufhin seine bisher geleistete Arbeit.[321] Als schließlich das 386BSD-Projekt weiterging, geriet man in juristischen Streit mit AT&T und durfte das System bis Ende 1993 nicht veröffentlichen.[322]
- Stallman selbst hatte sich 1988 mit seinem ***GNU-Projekt*** verrannt. Zwar waren die meisten der vielen kleinen Teile eines Unix-Systems fertig, was aber fehlte, war sein Kern. 1986 hatte Stallman noch gehofft, eine Entwicklung des MIT übernehmen zu können, doch die erwies sich als unbrauchbar. Zwei Jahre später entschied er sich für den sogenannten „Mach-Microkernel", eine damals moderne Technik, die aber schwer zu beherrschen war, wie Stallman inzwischen zugibt: „[...] it was unreliable. It took us years and years to get the GNU kernel to work. [...] So maybe that's one of the mistakes that I made: that design decision."[323] Fans, die auf ein freies Unix warteten, das sie auf den PC übertragen konnten, wurde dieser Kern mit dem Namen „Hurd" seit 1990 versprochen. Er ist heute noch in Arbeit.
- Wenigstens ein erschwingliches Unix für den PC gab es: Das 1987 veröffentlichte ***Minix.*** Der in Amsterdam lehrende amerikanische Informatik-Professor Andrew Tanenbaum hatte es in seinen Abendstunden geschrieben, zwei Jahre lang, als direkte Reaktion auf die Entscheidung von AT&T, die Weitergabe des Unix-Quellcodes an Studenten zu verbieten. Gemeinsam mit einem Lehrbuch wurde Minix für ca. 60 Dollar verkauft.[324] Als unzugänglich erwies sich Tanenbaum aber, wenn es um Veränderungen ging. Immer wieder wies er darauf hin, daß das System nur Lehrzwecken dienen und nicht produktiv genutzt werden sollte. Auch Anfragen Stallmans, ob Minix nicht die verbliebene Lücke im GNU-System füllen könnte,

---

[319] Das System war nicht im Quellcode verfügbar, konnte also nicht angepaßt werden. MS-DOS verfügte bis 1993 über keine Netzwerkfähigkeiten, sondern war auf ebenfalls proprietäre Software von Dritten angewiesen (Novell Network). Es ließ sich zwar GNU-Software portieren, diese litt aber unter dem komplizierten und instabilen Speichermodell von MS-DOS. Allerdings entwickelten Hacker auf MS-DOS Mailbox-Systeme und DFÜ-Anwendungen und schufen so unter anderem Zugangsmöglichkeiten zu den Foren im Internet (Usenet-Newsgroups); so daß der Satz „DOS eignete sich nicht zum Hacken" zumindest unter diesem Gesichtspunkt eingeschränkt werden muß.

[320] Wahrscheinlich war es die Firma „Berkeley Software Design Inc.", die das an der Universität entwickelte System vermarktete und verkaufte.

[321] Vgl. RAYMOND: The Art…, a.a.O.

[322] Das BSD-Projekt spaltete sich kurz danach weiter auf in NetBSD und FreeBSD; beide Systeme sowie der Abkömmling OpenBSD existieren heute noch und haben einen eigenen Kreis von Fans. Auch Mac OS X beruht teilweise auf BSD.

[323] STALLMAN: Free Software: Freedom…, a.a.O., S. 172.

[324] Vgl. MOODY, a.a.O., S. 32f

beschied Tanenbaum negativ.[325] Da er zwar den Quellcode veröffentlicht, aber nicht freigegeben hatte, war das System eine Sackgasse für Hacker – was sie aber nicht hinderte, es zu verwenden: Minix hatte bald mehr als 40.000 Nutzer.

„Welcher Bastler kann es sich erlauben, drei Mannjahre in das Programmieren, die Fehlersuche und die Dokumentation zu stecken und anschließend alles zu verschenken?“, lautete die rhetorische Frage in Gates’ „letter to hobbyists“. Tanenbaum gab die suggerierte Antwort: Keiner. Er fürchtete den Verlust der Kontrolle über Minix. Die Unix-Unternehmen dachten ähnlich. Stallman dagegen hatte die Frage, wer sich das leisten könne, unerwartet beantwortet: Ein Idealist mit altruistischen Motiven. Doch Stallman hatte mit seinen Plänen die Bodenhaftung verloren. Doch es gab noch eine weitere Antwort: Ein Student.

3.2 Linux

Von 1986 bis 1994 gab es also ein Zeitfenster für die Entwicklung eines Unix für Personal Computer.[326] Und weil diese PCs sich rasant verbreiteten, verstärkte sich in dieser Zeit auch die Nachfrage nach einem solchen Unix. Außerdem reifte ein Instrument heran, das erstmals eine ortsunabhängige, verteilte Produktion und deren Koordination ermöglichte: das Internet. Damals hatte es nicht die Form des „World Wide Webs“, mit dem der Begriff heute assoziiert wird, sondern bot vor allem E-Mail, Diskussionsforen (Usenet-Newsgroups) und FTP,[327] also Orte im Netz, auf denen man Dateien zur Verfügung stellen oder bekommen konnte.

Weitere Hilfsmittel, wie die freie Software „patch“[328], ermöglichten es Programmierern, das Internet auch praktisch zu einer Zusammenarbeit zu nutzen und auch die technische Basis war vorhanden: Dank einer regen Mailbox- und DFÜ-Szene waren inzwischen Modems erhältlich, die zwar nicht schnell (2400 baud[329]), aber verhältnismäßig preiswert[330] waren und so den privaten Anschluß an das Internet ermöglichten.

Die Methode, gemeinsam über das Internet an Software zu arbeiten, war damals zwar nicht weit verbreitet, aber auch nicht unbekannt:[331] Beispiele sind die Programmiersprache „Perl“, die bereits kurz nach ihrer Erstveröffentlichung 1987 von ihrem Nutzerkreis weiterentwickelt wurde[332] oder der Webserver „Apache“, der 1994 unabhängig von GNU/Linux entstand. Auch der Informationsaustausch der Minix-User in ihrem Internet-

---

[325] „‘He approached me a couple of times,’ Tanenbaum says, ‘but he had all these conditions; it's got to be this, and it's got to be that. I said »whoa, whoa, whoa.“ He's kind of an abrasive person and it didn't really turn me on.’“ Ebd., S. 32.

[326] Zeitfenster (engl. „window of opportunity“) bedeutet in diesem Zusammenhang schlicht die Abwesenheit eines Produkts, für das sich in dieser Zeit eine Nachfrage entwickelte.

[327] Akronym für „File Transfer-Protocol“.

[328] Das Programm „patch“ (engl. für: „flicken“) ermöglicht es, bei der gemeinsamen Arbeit an einem Quellcode einzelne verbesserte Teile leichter austauschen und einflicken zu können. Vgl. RAYMOND: The Art…, a.a.O.

[329] Das bedeutet, für den Download dieser, etwa 1 MB großen Studienarbeit würden 58 Minuten benötigt.

[330] 1993 kostete ein 2.400 baud-Modem (ESCOM Data Hawk) 299,- DM.

[331] Man findet gelegentlich die Behauptung, Linus Torvalds habe mit Linux vor allem eine Entwicklungsmethode erfunden. Mit den Beispielen Perl und Apache soll deutlich gemacht werden, daß dies nicht korrekt ist – allerdings gewann diese Methode aufgrund von Linus Torvalds’ Führungsstil erheblich an Popularität.

[332] MOODY, a.a.O., S. 135-137.

Diskussionsforum „comp.os.minix“ bot einen Ansatz dazu, auch wenn Minix-Schöpfer Tanenbaum daraus weit weniger Nutzen zog als einer seiner Nutzer.

3.2.1 Linus Torvalds und die Entstehung von Linux

Abbildung 18: Linus Torvalds im Jahr 2002.[333]

Das Bedürfnis des finnischen Studenten Linus Benedict Torvalds nach einem eigenen Unix zu Hause erinnert an die „Hands-On Imperative“ der ersten Hacker. Torvalds hatte 1988 mit 19 Jahren ein Studium der Informatik (sowie im Nebenfach Physik und Mathematik) an der Universität Helsinki begonnen. Nach seinem Wehrdienst[334] hatte er sich im Sommer 1990 mit Tanenbaums Buch „Operating Systems: Design and Implementation“[335] auf sein zweites Studienjahr vorbereitet.[336] Seine Begeisterung für das Seminar „C und Unix“ sowie seine ersten Erfahrungen am neu erworbenen Unix-Computer der Universität wurden allerdings durch lange Wartezeiten getrübt.[337] Im Januar 1991 kaufte Torvalds einen PC, um darauf Minix zu installieren und Unix unabhängig von den Einschränkungen durch die Universität besser kennenzulernen.[338]

Sein erstes Software-Projekt begann Torvalds, um sich von zu Hause aus per Modem mit dem Unix-Rechner der Universität verbinden zu können und Diskussionsforen im Usenet zu lesen.[339] Dabei handelte es sich wohl vor allem um fachspezifische Foren wie

---

333 Foto aus: McMILLAN, Robert: The Great Dictator. Linus Torvalds: The Benevolent, Brilliant Keeper of the Kernel. In: LINUX Magazine, San-Francisco: 12/2002. Online veröffentlicht unter: http://www.linux-mag.com/2002-12/linus_01.html [Aufgerufen am 10.01.2005]. Das Foto wurde urheberrechtlich freigegeben unter einer Creative Commons Lizenz. Siehe auch: http://commons.wikimedia.org/wiki/Image:Linus_Torvalds.jpeg [Aufgerufen am 10.01.2005].

334 Nach seinem ersten Studienjahr leistete er von 1989 bis Mai 1990 seinen elfmonatigen Wehrdienst ab; deshalb begann sein zweites Studienjahr erst im Herbst 1990.

335 Es handelt sich um den gleichen Tanenbaum, der Minix geschrieben hatte; Minix wird in diesem Buch ausführlich besprochen. Vgl. TANENBAUM, Andrew Stuart: Operating Systems: Design and Implementation. Englewood Cliffs, N.J.: Prentice-Hall, 1987.

336 Dies erfolgte sozusagen im „Trockenschwimmen"; Torvalds hatte zunächst keinen Unix-fähigen Computer.

337 Der Computer an der Universität von Helsinki erlaubte nur den Zugriff von 16 Benutzern gleichzeitig. Vgl. MOODY, Glyn: Rebel Code. Linux and the Open Source Revolution. New York: Penguin Books, 2002, S. 31.

338 Vgl. TORVALDS: Just for…, a.a.O., S. 59-71.

339 Vgl. MOODY, a.a.O.,. 38f.

„comp.os.minix“ und möglicherweise auch um Diskussionen zu Science-Fiction-Themen.[340] Er beschloß, zu diesem Zweck einen sogenannten „Terminalemulator“ zu schreiben, da er mit den diesbezüglichen Fähigkeiten von Minix unzufrieden war. Tanenbaums Betriebssystem Minix sowie Stallmans GNU Compiler „gcc“ bildeten die Entwicklungsumgebung, in der Torvalds seine Software programmierte.[341] Der Terminalemulator selbst lief dagegen völlig unabhängig von Minix.

Das in Assemblersprache geschriebene Programm startete („bootete“) beim Einschalten des PCs von Diskette. Zunächst beherrschte es nur die Verbindung mit dem Universitätsrechner per Modem sowie die Eingabe und Ausgabe von Text – praktisch konnte man sich also nur anmelden und Emails oder Forenbeiträge lesen und beantworten. Allerdings genügte Torvalds diese Leistung im Frühjahr 1991 nicht mehr – er wollte auch Dateien auf seinem PC speichern können: Wenn zum Beispiel im Minix-Forum Verbesserungen diskutiert wurden und die betreffenden E-Mails längere Zitate von Quellcode („patches“) enthielten, wollte Torvalds diese Mails auch auf seinem PC abspeichern können (und sie nicht etwa per Hand abschreiben).

Also entwickelte er das ganze Frühjahr und den halben Sommer hindurch für das Terminalprogramm die Instrumente („Treiber“) zum Ansteuern seiner Festplatte und von Dateien darauf. Während der Fertigstellung im Juli 1991 wurde Torvalds dann klar, daß ein Programm, das direkt beim Hochfahren des PCs gestartet wird und das in der Lage ist, Dateien auf Festplatten zu lesen und zu schreiben, nichts anderes ist als ein Betriebssystem – ein unixartiges sogar, denn Torvalds orientierte sich bei der Programmierung des Systemverhaltens an Minix und dem Unix-System der Universität. Der Terminalemulator wurde zu „Linux“.[342]

Die Theorien über die Motive, die zur Entwicklung von Linux führten reichen von Philanthrophie bis zu purem Egoismus. Zusammenfassend ist zu sagen, daß die Idee für den Terminalemulator (und damit für Linux) zwar aus persönlichen Bedürfnissen entstand, aber nicht aus zwingender Notwendigkeit. Auch Minix verfügte über entsprechende Fähigkeiten und Torvalds hätte auch zurück zu DOS gehen können; für Microsofts Betriebssystem gab es damals bereits Terminalprogramme wie KERMIT. Das persönliche Bedürfnis (engl.: „personal itch“), das Raymond als Motiv anführt,[343] ist also nur der Anlaß für Torvalds Leistung gewesen. Als Ursache gibt er selbst spielerische Neugier und das Bedürfnis nach Weiterbildung an:

> „Programmieren ist eine Übung in Kreativität. Ich bin überhaupt erst zum Programmieren gekommen, weil ich herausfinden wollte, wie der Computer arbeitet. Zu meiner größten Freude entdeckte ich, daß es sich mit Computern verhält wie mit der Mathematik: Du kannst dir deine eigene Welt mit ihren eigenen Regeln aufbauen. [...] Die

---

[340] Obwohl Torvalds kulturell aus einer politisch eher linken Familie von Journalisten stammt, interessierte er sich selbst wenig für Politik. Als typischer „Geek“ seiner Zeit las Torvalds Douglas Adams’ „Per Anhalter durch die Galaxis“ und verbrachte Zeit mit Videospielen wie „Prince of Persia“, „Doom“ und später „Quake“. Vgl. TORVALDS: Just for…, a.a.O., S. 36 und WIRZENIUS, Lars: Linux Anecdotes. Rede für die Linux Expo. 27.04.1998. Online veröffentlicht unter: http://liw.iki.fi/liw/texts/linux-anecdotes.html [Aufgerufen am 18.12.2004].

[341] Vgl. TORVALDS: Just for…, a.a.O., S. 69f.

[342] Vgl. ebd., S. 86.

[343] RAYMOND vertritt in seinem 1997 erstmals veröffentlichten Vortrag „The Cathedral and the Bazaar“ die These, daß jede Software einem persönlichen Bedürfnis ihres Entwicklers entspringt. „Every good work of software starts by scratching a developer's personal itch“. RAYMOND: The Cathedral…, a.a.O., S. 23.

> Mathematik braucht sich keiner Beschränkung durch eine äußere Logik zu unterwerfen, aber sie muß in sich selbst und aus sich selbst heraus logisch sein. Wie jeder Mathematiker weiß, kann man eine Menge mathematischer Gleichungen aufstellen, bei denen drei plus drei gleich zwei ist. Man kann alles tun, was man will, aber je mehr Komplexität man hinzufügt, desto sorgfältiger muß man darauf achten, Inkonsistenzen mit der geschaffenen Welt zu vermeiden. Wenn das Geschaffene schön sein soll, darf es keine Fehler enthalten. Das ist das Wesen des Programmierens. [...] Mit Computern und Programmierung kannst du neue Welten entwickeln und manchmal entstehen dabei wunderschöne Muster."[344]

Neben der „Hands-On Imperative" der ersten Hacker-Regel („herausfinden, wie der Computer arbeitet") lassen sich aus Torvalds Darstellung seiner Beweggründe weitere Grundüberzeugungen von Hackern identifizieren: Seine Bewertung von Programmen als „schön" verweist auf die fünfte Regel der Hacker-Ethik, wonach Programmieren eine Kunstform sein kann. Seine Betonung der Unabhängigkeit von äußeren Regeln und der Möglichkeit, alles zu tun, „was man will" läßt sich auf die dritte Hacker-Regel zurückführen, die Ablehnung fremder Kontrolle. Diesen Willen zur Macht macht Torvalds in einem weiteren Statement deutlich:

> „Innerhalb der Grenzen deines Computers bist du der Schöpfer. Irgendwann bist du soweit, daß du das Geschehen komplett unter Kontrolle hast. Wenn du gut genug bist, kannst du Gott sein. Jedenfalls in bescheidenem Rahmen."[345]

Geträumt wurden diese Vorstellungen vom „Gott sein" in einer Welt, der sich Torvalds scheinbar entzogen hatte. Er spricht von wochenlangen „Marathon-Programmiersitzungen"[346] die nur aus „Programmieren, Schlafen, Programmieren, Schlafen, Programmieren, Essen"[347], bestanden. Er berichtet von der Gefahr, zu einem „Einsiedler"[348] zu werden und von „Hemmungen wegen meiner mangelnden sozialen Kompetenz"[349]. Letztere Einschätzung sollte sich als falsch erweisen. Auch der Mythos vom weltabgeschiedenen Programmiergenie, vergleichbar dem „Philosophen auf dem Berge", läßt sich nicht halten, wenn man bedenkt, welchem Zweck der Terminalemulator diente: Dem Lesen und Verschicken von E-Mail, d.h. der Kommunikation. In der virtuellen Gemeinschaft der Minix-Nutzer hatte Torvalds in dieser Zeit bereits einen Platz eingenommen – mit seiner Freigabe des „Terminalemulators" alias Linux rückte er sich dort in den Mittelpunkt.

3.2.2 Die Freigabe von Linux

Seine Erkenntnis, daß er dabei war, nicht nur ein Terminalprogramm, sondern ein Betriebssystem zu schreiben, publizierte Linus Torvalds am Sonntag, den 25.08.1991, im Minix-Forum:

---

344 TORVALDS: Just for…, a.a.O., S. 82.
345 Ebd., S. 81f.
346 Ebd., S. 86.
347 Ebd.
348 Ebd., S. 85.
349 Ebd.

> “Von: Linus Benedict Torvalds (torvalds@klaava.Helsinki.FI)
> Betrifft: What would you like to see most in minix?
> Newsgroups: comp.os.minix
> Datum:1991-08-25 23:12:08 PST
>
> Hello everybody out there using minix – I’m doing a (free) operating system (just a hobby, won’t be big and professional like gnu) for 386(486) AT clones. This has been brewing since April, and is starting to get ready. I’d like any feedback on things people like/dislike in minix, as my OS resembles it somewhat (same physical layout of the file-system (due to practical reasons) among other things).
>
> I’ve currently ported bash(1.08) and gcc(1.40), and things seem to work. This implies that I’ll get something practical within a few months, and I’d like to know what features most people would want. Any suggestions are welcome, but I won’t promise I’ll implement them :-)
>
> Linus (torvalds@kruuna.helsinki.fi)
>
> PS. Yes – it’s free of any minix code, and it has a multi-threaded fs. It is NOT protable (uses 386 task switching etc), and it probably never will support anything other than AT-harddisks, as that’s all I have
> :-(.”[350]

Torvalds kündigte das neue Betriebssystem als „frei“ an, wobei „frei“ nicht nur bedeutete, daß er beabsichtigte, den Quellcode offenzulegen, sondern auch, daß er das Programm kostenlos abgeben wollte. Torvalds verstand unter freier Software zunächst vor allem Gratis-Software, wie in seiner späteren E-Mail-Auseinandersetzung mit Minix-Erfinder Tanenbaum deutlich wurde:

> “[...] look at who makes money off minix, and who gives linux out for free. Then talk about hobbies. Make minix freely available, and one of my biggest gripes with it will disappear. Linux has very much been a hobby (but a serious one: the best type) for me: I get no money for it, and it’s not even part of any of my studies in the university.”[351]

In seiner ersten Mail zeigt Torvalds außerdem Respekt vor dem GNU-Projekt und erwähnt, daß er Bestandteile dieses Projekts für sein System einsetzt.[352] Im Hinblick auf seine Ziele äußert er sich bescheidener als Stallman bei dessen Bekanntgabe des GNU-Projekts: Das System werde nicht so groß und professionell und strebe lediglich an, eine Nachbildung von Minix zu sein.

Linus Torvalds beabsichtigte zu diesem Zeitpunkt, sein Programm „Freax“ zu nennen, eine Zusammensetzung aus „Free“ für „Free Software“, dem Wort „Freak“ und dem Unix-„X“.[353] Dieser Name mißfiel jedoch dem Administrator der Universität von Helsinki:

---

[350] TORVALDS, Linus Benedict: What would you like to see most in minix? Mail an die Newsgroup comp.os.minix vom 25.08.1992 23:12:08 Uhr PST. Online archiviert unter: http://groups.google.com/groups?hl=de&lr=&threadm=1991Aug25.205708.9541%40klaava.Helsinki.FI&rnum=1&prev=/groups%3Fselm%3D1991Aug25.205708.9541%2540klaava.Helsinki.FI [Aufgerufen am 06.01.2005].

[351] The Tanenbaum-Torvalds Debate. In: DIBONA, Chris; OCKMAN, Sam; STONE, Mark (Hsg.): Open-Sources. Voices from the Open Source Revolution. Sebastopol: O'Reilly, 1999, S. 223 (Appendix A).

[352] Er nennt „bash“ (einen Befehlsinterpreter) und GCC (Abkürzung für: GNU C Compiler; später: „GNU Compiler Collection“). Torvalds bediente sich der GNU-Programme bei der Entwicklung und dem Test seines Betriebssystem-Kerns.

[353] Vgl. WIKIPEDIA (deutsch): Geschichte von Linux. In: http://de.wikipedia.org/wiki/Geschichte_von_Linux [Aufgerufen am 08.01.2005].

Er benannte das öffentlich zugängliche Verzeichnis, das er Torvalds zur Verbreitung seines Programms auf dem Rechner der Universität einräumte, eigenmächtig als „ftp://ftp.funet.fi/pub/OS/Linux/“, wobei Linux für „Linus’ X-Server“ stehen sollte.[354] Torvalds akzeptierte diese Wahl, obwohl er den Namen zunächst als zu selbstgefällig abgelehnt hatte.[355]

Sowohl die Erwähnung des GNU-Projekts in der Ankündigungs-E-Mail als auch die Anlehnung des ursprünglich gewählten Namens „Freax“ an den Begriff „Free Software“ erwecken den Eindruck, daß Torvalds von Stallmans Philosophie beeinflußt war. Bestärkt wird das durch die Tatsache, daß Torvalds die technischen Anmerkungen[356] der einen Monat später veröffentlichten Version 0.01 seines Systems mit „Happy Hacking“ unterzeichnete – dem traditionellen Abschiedsgruß Stallmans. Torvalds hatte Stallman kurz zuvor[357] bei einer Veranstaltung an der Technischen Universität von Helsinki kennengelernt. Er erklärte später, Stallman habe ihn vor allem aufgrund seines äußeren Erscheinungsbildes beeindruckt; sein Vortrag sei „keine Erleuchtung für mich gewesen, aber etwas davon muß wohl hängen geblieben sein“.[358]

Auch wenn man die erwähnten Indizien für einen Einfluß Stallmans auf Torvalds nicht überbewerten sollte, bleibt festzustellen, daß sich der knapp 22jährige Torvalds der Netzgemeinschaft als „Freak“ darstellen wollte, als „Hacker“ in der Tradition Stallmans („Happy Hacking“), wenn auch mit weniger weitreichenden Ambitionen. So beschreibt Torvalds in einer E-Mail den Kernel 0.02: “This is a program for hackers by a hacker. I’ve enjoyed doing it, and somebody might enjoy looking at it and even modifying it for their own needs.”[359] Daß er sich als Teil der Hacker-Kultur sah, macht er auch für die Entscheidung verantwortlich, für Linux kein Geld zu verlangen:

> „Tatsächlich wollte ich das Geld aus einer ganzen Reihe von Gründen nicht. Als ich Linux erstmals ins Internet stellte, hatte ich das Gefühl, in die jahrhundertealten Fußstapfen der Wissenschaftler und Forscher zu treten, die ihre Arbeit auf den Grundfesten anderer aufsetzen – auf den Schultern von Giganten, um mit Isaac Newton zu sprechen. [...] Tatsache ist: Um Linux nutzbar zu machen, hatte ich mich auf eine Menge Tools verlassen, die frei über das Internet verteilt worden waren – ich hatte mich auf die Schultern von Giganten gehievt.“[360]

Eine dieser „Schultern“ war „gcc“, der von Stallman verfaßte C-Compiler des GNU-Projekts.[361] Weil er selbst in hohen Maße von dieser freien Software profitierte, entschloß sich Torvalds, Linux ebenfalls ab Version 0.12 unter der GNU GPL freizugeben:

---

354 Vgl. ebd.

355 Vgl. TORVALDS: Just for…, a.a.O., S. 97.

356 Engl.: „Release-Notes“. Die Release-Notes von Kernel 0.01 sind im Internet veröffentlicht unter: http://www.shortfamilyonline.com/tech/unix/history-of-linux/reference/RELNOTES-0_01.html [Aufgerufen am 08.01.2005].

357 Siehe Fußnote Nr. 295.

358 TORVALDS: Just for…, a.a.O., S. 66.

359 TORVALDS, Linus Benedict: Free minix-like kernel sources for 386-AT. Mail an die Newsgroup comp.os.minix vom 05.10.1991 09:24:25 PST. Online veröffentlicht unter anderem bei: http://www.chrisshort.net/archives/open-source-software/reference/05-Oct-1991-free-minix-like-kernel-sources-for-386-at.html {[Aufgerufen am 10.01.2005].

360 TORVALDS: Just for…, a.a.O., S. 103-105.

361 Vgl. TORVALDS, Linus: The Linux Edge. In: DIBONA u.a. (Hrsg.), a.a.O., S. 101-111, hier S. 107.

> "While I was nervous about the GPL at first, I also wanted to show my appreciation to the gcc C compiler that Linux depended on, which was obviously GPL'd. Making Linux GPL'd was definitely the best thing I ever did."[362]

Zuvor hatte er eine selbstgemachte Lizenz[363] verwendet, die jegliche kommerzielle Verbreitung kategorisch ausschloß, also sehr viel strenger war als die GPL. Der Lizenzwechsel bildete die Basis für den späteren wirtschaftlichen Erfolg des Betriebssystems GNU/Linux.

3.2.3 Die Entstehung der Linux-Community

Schon der Betreff von Torvalds erster Mail zeigte das Organisationsmodell, nach dem er sein System entwickeln wollte: „What would you like to see most in minix?" – die Frage an die Minix-Nutzergemeinschaft, was sie sich am meisten von ihrem Programm wünschen, macht deutlich, daß Torvalds Linux im Dialog mit anderen entwickeln wollte; völlig anders als Minix-Urheber Tanenbaum, der auf Anfragen ablehnend reagierte.

Torvalds erhielt auf diese Mail zwar durchweg positives Feedback, die Menge hielt sich jedoch zunächst in Grenzen: Nur fünf verschiedene Mitglieder des Minix-Forums antworteten Torvalds öffentlich, äußerten Wünsche, forderten mehr Informationen oder boten Hilfe an. In privater E-Mail erhielt Torvalds zwar weitere Reaktionen, etwa Angebote, das System zu testen. Aber selbst Mitte September 1991, als er Linux 0.01 fertiggestellt hatte, gab es nicht mehr als zehn bis fünfzehn Interessenten.[364]

Dies sollte sich erst mit Erscheinen der Version 0.12 am 05. Januar 1992 ändern. Wenige Wochen zuvor hatte Torvalds seinen eigenen Rechner vom Betriebssystem Minix komplett auf Linux umgestellt.[365] Auf Drängen von Nutzern entschied er dann, Linux die Fähigkeit zu geben, Daten aus dem Arbeitsspeicher zeitweilig auf die Festplatte auszula-

---

[362] YAMAGATA, Hiroo: The Pragmatist of Free Software. Linus Torvalds Interview. In: Wired Japan, 20.09.1997. Online veröffentlicht unter: http://kde.sw.com.sg/food/linus.html [Aufgerufen am 21.01.2005].

[363] Die Lizenz des Linux-Kernels bis Version 0.11 lautete:
"This kernel is (C) 1991 Linus Torvalds, but all or part of it may be redistributed provided you do the following:
- Full source must be available (and free), if not with the distribution then at least on asking for it.
- Copyright notices must be intact. (In fact, if you distribute only parts of it you may have to add copyrights, as there aren't (C)'s in all files.) Small partial excerpts may be copied without bothering with copyrights.
- You may not distribute this for a fee, not even „handling" costs."
MOODY, a.a.O., S. 44. Die vollständigen Release-Notes von Kernel 0.01 sind im Internet veröffentlicht unter:
http://www.shortfamilyonline.com/tech/unix/history-of-linux/reference/RELNOTES-0_01.html [Aufgerufen am 08.01.2005].

[364] Ebd., S. 43.

[365] Nicht ganz freiwillig – er hatte bei seinen Experimenten versehentlich einen Teil seiner Festplatte überschrieben. Er entschied dann, daß er – statt Minix neu zu installieren – sich komplett auf sein eigenes System verlassen wollte.

gern; ein „Page-to-Disk“ genanntes Verfahren.[366] Es war die erste Eigenschaft von Linux, die Torvalds nicht für sich selbst, sondern für andere entwickelte.[367] Mit dieser Eigenschaft von Version 0.12 erreichte der Linux-Kernels erstmals größere Verbreitung:

> „Ab diesem Punkt hob Linux ab. Plötzlich stiegen die Leute von Minix auf Linux um. Damals konnte Linux noch nicht alles, was Minix konnte, aber es konnte die meisten Sachen, auf die es den Leuten ankam. Und es besaß genau die eine Fähigkeit, auf die sie wirklich Wert legten. Mit Page-to-Disk konntest du größere Programme laufen lassen, als in deinen Speicher paßten. [...] In den ersten Wochen des Jahres 1992 war das ein Riesending. Es war im Januar als der Kreis der Linux-Anwender von fünf, zehn, zwanzig Leuten, die ich anmailen konnte und deren Namen ich kannte, auf Hunderte nicht mehr identifizierbare Leute anwuchs. Ich kannte nicht mehr jeden, der Linux einsetzte und das machte mir Spaß.“[368]

Die Diskussion und die Arbeit an Linux fand zu diesem Zeitpunkt noch im Minix-Forum statt, einem Bereich, der eigentlich der Arbeit an Minix dienen sollte. Es ist also kein Wunder, daß der Autor von Minix, Informatik-Professor Andrew Tanenbaum gereizt auf die zunehmenden Linux-Mails in „seinem“ Forum reagierte. Unter dem Betreff „Linux is obsolete“ begann er deswegen am 29. Januar 1992 eine öffentliche Diskussion, wobei er nicht nur das technische, sondern auch das soziale Konzept von Linus Torvalds massiv angriff.[369]

Abbildung 19 Andrew S. Tanenbaum, Professor an der Freien Universität Amsterdam.[370]

---

[366] Heutzutage beherrschen alle Betriebssysteme den Umgang mit „virtuellem Speicher“, wie jeder weiß, der während seiner Arbeit am PC gelegentlich die Festplatte klackern hört. Dabei verschiebt das Betriebssystem nicht verwendete Daten vom Arbeitsspeicher auf die Festplatte und lädt sie erst bei Bedarf wieder zurück – dadurch kommt es zum einen mit weniger Arbeitsspeicher aus, zum anderen kann es den Programmen vorgaukeln, es verfüge über mehr Arbeitsspeicher als physikalisch vorhanden. Die große Nachfrage nach dieser Fähigkeit erklärt sich aus der Tatsache, daß Arbeitsspeicher sehr viel teurer war und ist als Festplattenplatz. Im Falle von Linux 0.12 war es ein deutscher Student, der Torvalds um diese Fähigkeit bat, weil er sich mehr als 2 MB RAM Arbeitsspeicher nicht leisten konnte. Vgl. TORVALDS: Just for…, a.a.O., S. 100.

[367] Ebd., S. 100.

[368] Ebd., S. 102.

[369] Der Informatik-Professor bezeichnete dabei Linux öffentlich als technische Fehlentwicklung und brachte seine höhere Qualifikation mit den Worten zum Ausdruck, wenn Torvalds in seinem Seminar wäre, hätte er ihm eine schlechte Note gegeben. Torvalds antwortete entsprechend harsch und bezeichnete unter anderem Minix als „hirnlos“ ("brain-damaged“). Das Streitgespräch ist vollständig abgedruckt in: The Tanenbaum-Torvalds Debate, a.a.O. Online archiviert unter: http://www.chrisshort.net/archives/open-source-software/linux-is-obsolete/ [Aufgerufen am 10.01.2005].

[370] Quelle: Königliche Niederländische Akademie der Wissenschaften, 2004. Online veröffentlicht unter: http://www.knaw.nl/akademiehoogleraren/benoemd2004.html [Abgerufen am 10.01.2005].

Technisch kritisierte Tanenbaum die Architektur des Betriebssystems.[371] Bezüglich der sozialen Aspekte von Linux äußerte er Bedenken an Torvalds Bereitschaft, sein Werk unter der GPL freizugeben: Er könne nicht glauben, daß Torvalds die Kontrolle über das von ihm geschaffene System aus der Hand geben und zulassen wolle, daß jeder Linux nehmen und nach Belieben weiterentwickeln – oder ruinieren – könne.

> "An interesting question is whether Linus is willing to let LINUX become "free" of his control. May people modify it (ruin it?) and sell it? [...] Suppose Fred van Kempen returns from the dead[372] and wants to take over, creating Fred's LINUX and Linus' LINUX, both useful but different. Is that ok? The test comes when a sizable group of people want to evolve LINUX in a way Linus does not want."[373]

In einer weiteren öffentlichen E-Mail an die Minix-Nutzer zwei Tage später bezweifelte Tanenbaum die Möglichkeit, daß eine sich ständig verändernde Gemeinschaft von Freiwilligen ohne zentrale Kontrollinstanz über das Internet gemeinsam Software entwickeln könne. Tanenbaum verwies auf die damals vorherrschende wissenschaftliche Meinung, daß ein Software-Projekt genauso organisiert und geführt werden müsse wie ein Team bei einem chirurgischen Eingriff – Chefarzt und mehrere Assistenten – und nicht wie ein Mob:

> "I think coordinating 1000 prima donnas living all over the world will be as easy as herding cats [...] If Linus wants to keep control of the official version, and a group of eager beavers want to go off in a different direction, the same problem arises. I don't think the copyright issue is really the problem. The problem is co-ordinating things. Projects like GNU, MINIX, or LINUX only hold together if one person is in charge. During the 1970s, when structured programming was introduced, Harlan Mills pointed out that the programming team should be organized like a surgical team – one surgeon and his or her assistants, not like a hog butchering team – give everybody an axe and let them chop away. Anyone who says you can have a lot of widely dispersed people hack away on a complicated piece of code and avoid total anarchy has never managed a software project."[374]

Linus Torvalds antwortete einen Tag später auf diese Zweifel:

---

[371] Dabei ging es zum einen um die Übertragbarkeit von Linux auf andere Systeme als solche mit i386er-Prozessoren, zum anderen ging es um die grundsätzliche Frage, ob man ein Betriebssystem mit monolithischem Kernel oder mit Microkernel konstruieren sollte. Bei einem monolithischem Kernel kümmert sich ein einheitliches System um alle Aufgaben, bei einem Microkernel werden so viele Aufgaben wie möglich an spezialisierte Module vergeben. Letzterer Ansatz galt damals als flexibler, allerdings wurde der Kommunikationsaufwand zwischen den einzelnen Modulen weit unterschätzt. Linux ist noch heute als monolithischer Kernel realisiert.

[372] Fred van Kempen hatte sich zuvor mit Änderungen und Verbesserungen an Minix engagiert. Gegen Tanenbaums Wunsch – und ohne rechtliche Grundlage – wollte er aber sein verändertes Minix selbst veröffentlichen. Dies untersagte Tanenbaum, worauf sich van Kempen im Minix-Forum nicht mehr weiter einbrachte (deswegen: „returns from the dead"). Später beteiligte sich van Kempen an der Linux-Entwicklung, vor allem im Bereich der Netzwerkfähigkeiten, legte aber auch dabei ein eigenmächtiges Verhalten an den Tag. Vgl. auch: MOODY, a.a.O., S. 71-78.

[373] TANENBAUM, Andrew Stephen: Unhappy campers. Mail an die Newsgroup comp.os.minix vom 3 Feb 92 22:46:40 GMT. Online veröffentlicht unter: http://www.chrisshort.net/archives/open-source-software/linux-is-obsolete/ [Aufgerufen am 10.01.2005].

[374] TANENBAUM, Andrew Stephen: Re: Unhappy campers. Mail an die Newsgroup comp.os.minix vom 5 Feb 92 23:23:26 GMT. Online veröffentlicht unter: http://www.chrisshort.net/archives/open-source-software/linux-is-obsolete/ [Aufgerufen am 10.01.2005].

> "Here's my standing on "keeping control", in 2 words (three?):
> I won't." [375]

Seine Fähigkeit, die Entwicklungsrichtung von Linux zu bestimmen, sei lediglich dem Umstand geschuldet, daß er das System derzeit besser kenne als jeder andere, führte Torvalds in dieser E-Mail aus. Dies werde sich allerdings mit der Zeit ändern, wenn immer mehr Entwickler sich an der Arbeit am System beteiligten.[376] Er habe lediglich die Hoffnung, daß er die Entwicklung noch länger anführen könne, da sich noch niemand beschwert habe – rechtlich habe er aber keinen Anspruch darauf. Falls das Vertrauen in ihn verloren ginge, könne er jederzeit abgelöst werden.

> "That aside, if Fred van Kempen wanted to make a super-linux, he's quite welcome. He won't be able to make much money on it (distribution fee only), and I don't think it's that good an idea to split linux up, but I wouldn't want to stop him even if the copyright let me. [...] Yes, coordination is a big problem, and I don't think linux will move away from me as "head surgeon" for some time, partly because most people understand about these problems. But copyright /is/ an issue: if people feel I do a bad job, they can do it themselves."[377]

Es ist eine Folge der GNU General Public License, daß einmal freigegebene Software nicht mehr zurückgenommen werden kann. Zwar bleibt der ursprüngliche Urheber rechtlich bestehen; da er aber jedem das Recht verliehen hat, den Code zu modifizieren und weiterzugeben, hat er keine juristischen Sanktionsmöglichkeiten gegen mögliche Konkurrenten um die Führung seines Software-Projekts. Es kann also in der Tat jeder jederzeit eine eigene Version des Linux-Kernels weiterentwickeln und verbreiten – und falls diese den Zuspruch einer Mehrheit der Nutzer und Entwickler findet, wäre Torvalds selbst entmachtet.

Seine oben zitierte Mail – und seine Freigabe von Linux unter der GPL – drückt also auch ein großes Vertrauen in seine Nutzer und Mitprogrammierer aus, ein geradezu leichtinniges Vertrauen könnte man meinen, wenn man sich den heutigen Erfolg seines Betriebssystems vor Augen hält. Die Zweifel Tanenbaums erscheinen daher nachvollziehbar. Es ist aber festzustellen, daß Torvalds auch nach 14 Jahren noch immer anerkannter Leiter der Linux-Entwicklung ist; bei näherer Betrachtung wirkt sein damaliges Verhalten weniger altruistisch.

Tanenbaum reagierte auf das Feedback seiner Minix-Benutzer abweisend; Vorschläge zur Verbesserung des Programms lehnte er größtenteils ab, eigene Weiterentwicklungen

---

[375] TORVALDS, Linus Benedict: Re: Unhappy campers. Mail an die Newsgroup comp.os.minix vom 6 Feb 92 10:33:31 GMT. Online veröffentlicht unter: http://www.chrisshort.net/archives/open-source-software/linux-is-obsolete/ [Aufgerufen am 10.01.2005].

[376] Aus seinem Schreiben geht hervor, daß es bereits andere Mitarbeiter am Linux-Kernel gibt – er nennt den Programmierer Theodore Ts'o, der heute noch an der Entwicklung des Linux-Kernels beteiligt ist – und daß er mit weiteren rechnet, auch um dem System Eigenschaften hinzuzufügen, die er selbst mangels technischer Ausstattung nicht programmieren kann. Als Beispiel nennt er die Ansteuerung der (damals noch sehr teuren) SCSI-Geräte. Außerdem regt Torvalds die Schaffung eines eigenen Linux-Forums vor, auch um nicht weiter das Minix-Forum mit der Diskussion um Linux zu belasten.

[377] TORVALDS: Re: Unhappy campers, a.a.O.

seiner Nutzer verhinderte er durch Verweis auf sein Urheberrecht.[378] Aus der so düpierten Anwenderschar konnte keine Entwicklergemeinschaft entstehen. Bei Torvalds war es genau umgekehrt: Er strebte vor allem nach Reputation unter seinen Anwendern und Mitentwicklern, die rechtliche Kontrolle über den Code war ihm zweitrangig. „The power of Linux is as much about the community of cooperation behind it as the code itself“,[379] schrieb er 1999 und präzisierte diese Erkenntnis in seiner zwei Jahre später erschienen Autobiographie:

> „Linux hat mich nie aufgewühlt. Ich habe wegen Linux nie auch nur eine schlaflose Nacht verbracht. Was mich wirklich aufwühlte und immer noch aufwühlt, ist nicht die Technik an sich, sondern die soziale Interaktion, die damit verbunden ist. [...] Das soziale Ansehen, das ich bei diesen Leuten [im Minix-Forum] genoß, war mir wichtig [...] Eines der Dinge, die ich an Linux so schön und motivierend fand, war das Feedback, das ich bekam. Es bedeutete, daß Linux wichtig war, und war ein Zeichen, daß ich einer sozialen Gruppe angehörte. Die ich anführte. Das war mir wichtig, keine Frage.“[380]

Torvalds sieht sich also als Anführer einer sozialen Gruppe; seine ersten Anhänger rekrutierte er aus Tanenbaums Minix-Forum – zu dessen Ärger. Er war auch keineswegs bestrebt, diese Führungsrolle aufzugeben, auch wenn er mit der Freigabe von Linux unter der GPL diese Möglichkeit eröffnete. Anders als Tanenbaum war ihm aber klar, daß er das „soziale Ansehen“ in der Netzgemeinschaft nicht erzwingen konnte, sondern daß er es erwerben mußte.[381] Nur indem er sich dem Votum seiner Mitentwickler unterwarf, konnte er zu deren Anführer werden. Im folgenden wird diese Entwicklergemeinschaft des Linux-Kernels näher beschrieben.

3.2.4 Die Linux-Entwickler

Die meisten der frühen Entwickler des Linux-Kernels waren Studenten oder arbeiteten an Universitäten.[382] Linux war zunächst nur über das Internet zu bekommen; 1992 waren damit fast ausschließlich Hochschulen und große Firmen verbunden, nicht aber Privatleute.[383]

---

[378] Tanenbaum selbst begründet sein damaliges Verhalten inzwischen schlicht mit Überforderung: „I was going crazy with the endless stream of new features people were sending me. I kept refusing them all because I wanted to keep MINIX small enough for my students to understand in one semester. My consistent refusal to add all these new features is what inspired Linus to write Linux. Both of us are now happy with the results. [...] The excursion into hackerland was a detour.“ TANENBAUM, Andrew Stephen: Frequently Asked Questions. Veröffentlichung auf seiner Homepage unter: http://www.cs.vu.nl/~ast/home/faq.html [Aufgerufen am 19.01.2005].

[379] TORVALDS: The Linux…, a.a.O., S. 109.

[380] TORVALDS: Just for…, a.a.O., S. 119f.

[381] Dieses Streben nach Anerkennung erinnert an den Hacker-Grundsatz, daß man nur dadurch ein Hacker wird, indem man von anderen als Hacker bezeichnet wird: „You do not become a hacker by calling yourself a hacker – you become a hacker when others call you a hacker.“ RAYMOND: Homesteading the…, a.a.O., S. 94.

[382] Vgl. MOODY, a.a.O., S. 71.

[383] Prof. Dr. Friedrich KITTLER, Kulturwissenschaftler an der Berliner Humboldt-Universität, sieht eine enge Verbindung zwischen GNU/Linux und den Universitäten: „So direkt waren offene Quellen und freie Betriebssysteme an die Freiheit der Universität gekoppelt. Schauen Sie, wieviel ‘edu’ da drinsteht in den Linux-Kernel-Sources. So direkt hängt aber auch die Zukunft der Universität von diesen freien Quellen ab.“ KITTLER, Friedrich: Wissenschaft als Open-Source-Prozeß. Vortrag auf der Konferenz WOS1 (Wizards of OS – Offene Quellen und Freie Software) vom 16.-17.7.1999 im Haus der Kulturen der Welt, Berlin. Im Internet veröffentlicht unter: http://www.hydra.umn.edu/kittler/os.html [Aufgerufen am 18.1.2005].

Viele der ersten Mitarbeiter an Linux gehören noch heute zum inneren Zirkel um Torvalds: Beispielsweise ist Alan Cox[384], der bereits 1991 als 23jähriger an dem System mitarbeitete, noch immer Torvalds' „rechte Hand" bei der Kernelentwicklung; Theodore Ts'o[385] und Stephen Tweedie[386] sind ebenso lange dabei.

Abbildung 20: Einige der wichtigsten Linux-Entwickler bei dem „Linux 2.5 Kernel Summit" am 30.03.2001 in San Jose, California.[387]

Vor allem zu Beginn der Entwicklung von Linux waren – ähnlich wie in der Hackergemeinschaft am MIT – Anwender und Programmierer meistens identisch.[388] Das war auch ganz notgedrungen so, denn obwohl Torvalds bei der Vorstellung von Kernelversion 0.12 behauptet hatte: „Earlier releases were very much only for hackers"[389] so blieb doch auch später die Installation und Einrichtung eines GNU/Linux-Systems nicht eben trivial, wie ein Ausschnitt aus den Installationsanweisungen für Kernel 0.95 illustrieren mag.

> "Boot up linux again, fdisk to make sure you now have the new partition, and use mkfs to make a filesystem on one of the partitions fdisk reports. Write "mkfs -c /dev/hdX nnn" where X is the device number reported by linux fdisk, and nnn is the size – also reported by fdisk. nnn is the size in /blocks/, ie kilobytes. You should be able to use the size info to determine which partition is represented by which device name."[390]

---

384 Vgl. WIKIPEDIA (engl.): Alan Cox. In: http://en.wikipedia.org/wiki/Alan_Cox [Aufgerufen am 18.1.2005].

385 Der Informatiker und Absolvent des MIT, Theodore Ts'o arbeitete unter anderem an Authentifizierungssystemen von GNU/Linux sowie (bis heute) am Linux-Kernel. WIKIPEDIA (engl.): Theodore Ts'o. In: http://en.wikipedia.org/wiki/Theodore_Ts%27o [Aufgerufen am 18.1.2005].

386 Der schottische Programmierer Stephen Tweedie ist verantwortlich für das Linux-Dateisystem „ext3" und ist weiterhin an der Kernelentwicklung beteiligt. WIKIPEDIA (engl.): Stephen Tweedie. In: http://en.wikipedia.org/wiki/Stephen_Tweedie [Aufgerufen am 18.1.2005].

387 Foto aus: CORBET, J.: The Linux 2.5 kernel summit. LWN Feature Article. Linux Weekly News, 02.04.2001. Veröffentlicht unter: http://lwn.net/2001/features/KernelSummit/ [Aufgerufen am 18.01.2005] Zu sehen sind unter anderem Alan Cox (vorderste Reihe, zweiter von links), Theodore Ts'o (Mitte, zweite Reihe), Stephen Tweedie (letzte Reihe, dritter von links), Linus Torvalds (mittlere Reihe, sechster von links) und Eric Steven Raymond (mittlere Reihe, achter von links). Eine vollständige Bildbeschreibung findet sich auf: http://lwn.net/2001/features/KernelSummit/annotated.php3.

388 Vgl. ETTRICH. Matthias: Koordination und Kommunikation in Open-Source-Projekten. In: GEHRING, Robert A.; LUTTERBECK, Bernd (Hsg.): Open Source Jahrbuch 2004. Zwischen Softwareentwicklung und Gesellschaftsmodell. Berlin: Lehmanns Media, 2004, S. 79-192, hier S. 185.

389 MOODY, a.a.O., S. 48.

390 TORVALDS, Linus Benedict: Release notes for Linux v0.95. Veröffentlicht am 07.03.1992. Online abrufbar unter anderem unter: http://source.rfc822.org/linux/kernel/Historic/old-versions/RELNOTES-0.95 [Aufgerufen am 19.01.2005]. Owohl die Installation heute sehr viel einfacher ist, so erschließen sich immer noch viele Einstellungen unbedarften Anwendern nicht.

Die Kommunikation zwischen den Linux-Entwicklern findet bis heute hauptsächlich per E-Mail statt: Zunächst im Usenet in der Minix-Newsgroup, dann, nach der Auseinandersetzung mit Tanenbaum, in einem eigenen Linux-Forum unter „alt.os.linux“, später „comp.os.linux“.[391] Neben Newsgroups wurden auch Mailinglisten verwendet: Die erste, „Linux-activists“, wurde im Oktober 1991 gegründet und hatte bereits im folgenden Februar 400 Teilnehmer.[392] Zunächst beschränkte sich die Zusammenarbeit auf „tägliche E-Mails einer Handvoll Geeks“[393]; sie enthielten Patches (also kleinere Quellcode-Teile mit Fehlerkorrekturen oder neuen Funktionen) oder einfach nur Feedback.

Um die Zusammenarbeit technisch zu erleichtern, paßte Torvalds die Struktur des Linux-Kernels an:[394] Er wurde grundsätzlich modular aufgebaut, mit klar definierten Schnittstellen, damit die Komponente des einen Entwicklers nicht durch die eines anderen in Mitleidenschaft gezogen werden konnte.

> “With the Linux kernel it became clear very quickly that we want to have a system which is as modular as possible. The open-source development model really requires this, because otherwise you can’t easily have people working in parallel. [...] The key is to keep people from stepping on each other’s toes.”[395]

Die Technik ermöglichte es also jedem einzelnen, autonom zu arbeiten, und sich als Experte eines bestimmten Arbeitsgebiets zu profilieren. Dies wird auch durch die Philosophie der Unix-Programmierung unterstützt: Aufgaben werden nicht von einem großen Programm gelöst, sondern von vielen kleinen, die jeweils nur für eine Teillösung zuständig sind, aber über klar definierten Schnittstellen beliebig kombiniert werden können.[396] Für die Arbeit innerhalb der Linux-Gemeinschaft heißt das ganz äquivalent, daß man einzeln arbeitet, aber seine Ergebnisse für andere verwendbar macht.

Die Möglichkeit, parallel an verschiedenen Aspekten der Software zu arbeiten, war bereits in der Anfangsphase wichtig, denn die Entwicklung schritt rapide voran, mit mehreren veröffentlichten Versionen des Linux-Kernels im Monat. Auch dessen Eigenschaften wurden immer komplexer, etwa die Zusammenarbeit mit der freien grafischen Oberfläche „X“[397], die Netzwerkfähigkeiten[398] oder Übertragungen des Kernel-Codes auf andere Pro-

---

391 Außerdem wird per privater E-Mails kommuniziert sowie gelegentlich face-to-face bei internationalen Entwicklertreffen.

392 Vgl. MOON, Jae Yun; SPROULL, Lee: Essence of distributed work: The case of the Linux kernel. In: First Monday, Ausgabe 5, Nr. 11. November 2000. Online veröffentlicht unter: http://www.firstmonday.org/issues/issue5_11/moon/index.html [Aufgerufen am 21.01.2005].

393 TORVALDS: Just for…, a.a.O., S. 130. Das Wort „Geek“ bedeut so viel wie Technikfan oder Computerfreak.

394 Die lose Organisation der Linux-Entwickler über das Netz bestimmte also den Aufbau der Software. Diese Tatsache mag für die wissenschaftliche Organisationsforschung interessant sein, in der es bisweilen umstritten ist, ob Organisation Technik prägt oder umgekehrt Technik Organisation. Vgl. KIESER, Alfred; KUBICEK, Herbert: Organisation. 3. völlig neu bearbeitete Auflage. Berlin; New York: de Gruyter: 1992, S. 202f.

395 TORVALDS: The Linux…, a.a.O., S. 108.

396 Vgl. GANCARZ, a.a.O., S. 17ff.

397 Im Frühjahr 1992 begann die Portierung von X nach GNU/Linux durch Orest Zborowski, später übernahm Dirk Hohndel das Projekt. Hohndel war später einer der Gründer der deutschen Firma Suse Linux. Vgl. MOODY, a.a.O., S. 62f.

398 Im Oktober 1992 enthielt der Linux-Kernel erstmals Netzwerkfähigkeiten, Entwickler waren Fred van Kempen, später Alan Cox. Vgl. ebd., S. 72-77.

zessoren wie Sun SPARC und DEC Alpha.[399] Jede dieser Fähigkeiten eröffnete dem System wiederum neue Einsatzgebiete, so daß sich die Komplexität des Codes und die Zahl der Entwickler gegenseitig positiv beeinflußten:

> "The better Linux became, the more people used it; and the more people debugged it, the faster it improved: a virtuous circle that continues to drive Linux development at a vertiginous rate."[400]

Den parallelen Anstieg der technischen Komplexität und der Zahl der Entwickler zeigen die beiden folgenden Diagramme.

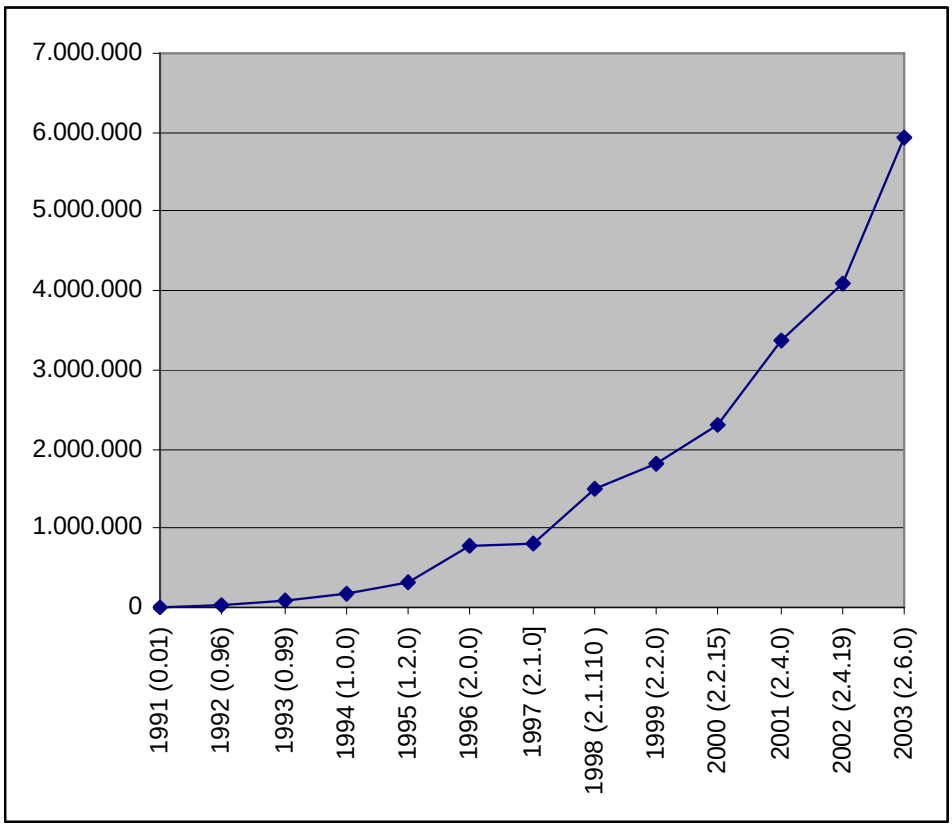


Diagramm 1: Die Zahl der Codezeilen im Linux-Kernel. Entwicklung von 1991 bis 2003.[401]

---

[399] Im November 1995 wurde die erste Portierung des Linux-Kernels durch Dave Miller abgeschlossen, Prozessor war der Sun SPARC. Torvalds selbst veränderte zur gleichen Zeit Linux so, daß es auf den Alpha-Prozessoren von DEC lief. Vgl. ebd., S. 108-111.

[400] Ebd., S. 60.

[401] DIEDRICHS weist darauf hin, daß ein Großteil des Kernel-Quellcodes (ca. 60 Prozent) aus Treiber-Software besteht; also Programmen zur Ansteuerung von Hardware. Sein Schluß, der Anteil der Treiber werde immer größer, läßt sich anhand der von ihm publizierten Grafik allerdings nicht belegen. DIEDRICH, Oliver: Happy Birthday, Tux! Zehn Jahre Freies Betriebssystem. In: c't Magazin für Computertechnik. Ausgabe 19/2001. Hannover: heise, 10.09.2001, S. 166. Online abrufbar unter: http://www.heise.de/ct/01/19/162/default.shtml [Aufgerufen am 18.01.2005].
Zu beiden Diagrammen sind die zugrundeliegenden Daten im Anhang aufgeführt.

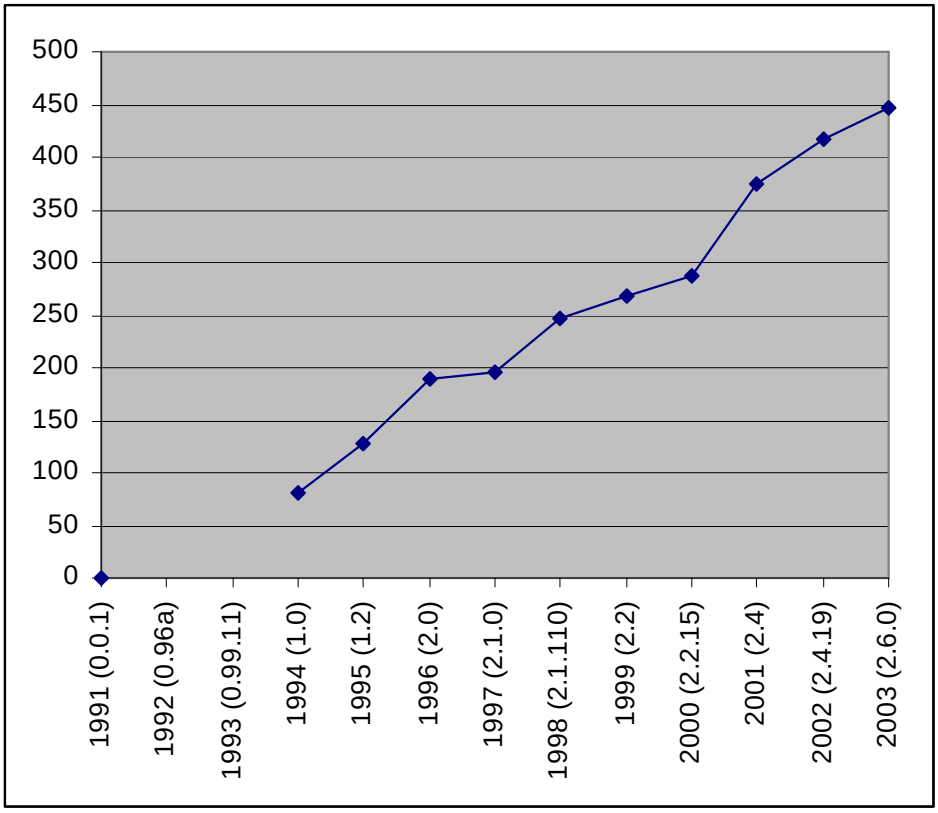


Diagramm 2: Besonders bedeutende Mitarbeiter am Linux-Kernel von 1991 bis 2003.[402]

Über die Anzahl der Linux-Entwickler gibt es keine exakten Angaben. Für *Abbildung 7* wurden die Namen in der sogenannten „Credits-Datei“ gezählt, einer Namensliste, die zusammen mit dem jeweiligen Kernel-Quellcode verteilt wird. Ursprünglich diente diese Liste Torvalds dazu, verdiente Mitstreiter auszuzeichnen; inzwischen fügt er dort auf Wunsch alle hinzu, die sich selbst für erwähnenswert halten.[403] Da die Credits-Datei von sehr vielen Interessierten, Anwendern und Programmierern gelesen wird, dürfte der soziale Druck aber dazu führen, daß die Liste eher zu kurz als zu lang ist.

Schätzungen zur Zahl der Entwickler reichen von mehreren hundert bis zu mehr als 40.000[404] – je nachdem, was bereits als Entwicklerbeitrag gezählt wird und ob man nur die Kernel-Entwicklung, das ganze GNU/Linux-Betriebssystem oder sogar freie Anwendungsprogramme mit einbezieht. DEMPSEY et al. gehen von 2429 Linux-Entwicklern im Jahr 2002 aus, schränken aber ein, daß die Mehrheit von ihnen nur jeweils ein bis zwei Beiträge geleistet habe.[405]

Die grundsätzliche Offenheit der Gruppe der Linux-Entwickler führt dazu, daß nicht nur ihre Zahl, sondern auch ihre soziale Zusammensetzung ständigen Schwankungen unterworfen ist. Zunächst wurde die Arbeit am Linux-Kernel vor allem von Freiwilligen vo-

---

[402] Nach der „Credits“-Namensliste. Zu Versionen unter 1.0 fehlen die Daten in der Grafik, da diese Versionen keine „Credits“-Datei enthalten. Für Version 0.0.1 gilt Torvalds Aussage, er habe den Quellcode ganz allein verfaßt. TORVALDS: Just for…, a.a.O., S. 95.

[403] Vgl. MOON; SPROULL, a.a.O.

[404] Ebd.

[405] Vgl. DEMPSEY, Bert J.; WEISS, Debra; JONES, Paul; GREENBERG, Jane: Who is an open source software developer? Communications of the ACM [Association for Computing Machinery], Vol. 45, Nr. 2, February 2002, S. 67-72. Zitiert nach: LUTHIGER, Benno: Alles aus Spaß. Zur Motivation von Open-Source-Entwicklern. In: GEHRING; LUTTERBECK (Hsg.), a.a.O., S. 93-105, hier S. 103.

rangetrieben. ROBLES u.a. haben 2001 ermittelt, daß 79 Prozent der Programmierer von freier Software für ihre Arbeit nicht bezahlt werden; rund 30 Prozent kommen aus dem universitären Bereich.[406] Das mag für viele dieser Projekte auch weiterhin gelten[407] – für die Arbeit am Linux-Kernel aber nicht: So hat einer der dafür Hauptverantwortlichen, Andrew Morton, im Jahr 2004 festgestellt, daß sich die soziale Zusammensetzung seiner Kollegen drastisch verändert hat, vor allem was den Anteil der Freizeit-Programmierer angeht.

> "Your typical stereotype is a male computer geek working in his basement writing code in his spare time, purely for the love of his craft. And certainly it is the case that such people were a significant force up until about five years ago. But nowadays, most of the core open source software developers are full-time employed professionals and contributing to public software projects is part of their job description.
> We take the example of the Linux kernel project. Over the past three years, there have been 38,000 changes to that tree – 38,000 pages. One thousand contributors, one thousand individuals have made those 38,000 changes. But 20 contributors have made half of those changes. So the remaining half of 38,000 lines came from the other 980 people. So it's a very non-uniform distribution of contribution rights there.
> If we take a look at where these people come from geographically, they mainly come from North America and Europe[408] and increasingly Eastern Europe. Australia is disproportionately well represented amongst the developers, but we have really very few developers from Asian countries although that is increasing. There is also a lot of open source activity in Latin America, although we're not seeing a proportional number of contributors at the source code level at this stage from Latin America. [...]
> Many of the major contributors, particularly to the Linux kernel, work for the Linux distributors – most prominently Red Hat and Suse. [...] Some of the kernel contributors are employed by hardware companies who wish to ensure that Linux works as well as possible on their latest products. Most prominent of those would be Intel, IBM, Hewlett-Packard, SGI, and others.
> Practically all of open source development is performed via e-mail on public mailing lists in which any interested individual can participate. This is very good for non-English speakers as well because they can absorb and produce communication at a speed which is comfortable for them. [...] We don't use telephones."[409]

---

406 Vgl. ROBLES, Gregorio; SCHEIDER, Hendrik; TRETKOWSKI, Ingo; WEBER, Niels: Who Is Doing It? A research on Libre Software developers. Berlin: Technische Universität Berlin, Fachgebiet für Informatik und Gesellschaft, August 2001. Online veröffentlicht unter: http://widi.berlios.de/paper/study.html [Aufgerufen am 18.01.2005].

407 So sieht BRANDT den typischen Programmierer der graphischen Linux-Oberfläche KDE als männlichen Studenten und Single zwischen 20 und 30 Jahre, der etwa 15 Stunden pro Woche unentgeltlich am KDE-Projekt mitarbeitet. BRANDT, Andreas: Structure and Organisation of KDE, KDE Contributors Conference 2003. Online veröffentlicht unter: http://events.kde.org/info/kastle/presentations/Presentation_Andreas_Brand_KDE_Nove_Hrady_24-09-2003.pdf [Aufgerufen am 18.01.2005]. Zitiert nach: BRUCHERSEIFER, Eva: Die KDE-Entwicklergemeinde – wer ist das? In: GEHRING; LUTTERBECK (Hsg.), a.a.O., S. 65-82, hier S. 78.

408 Nach übereinstimmenden Erkenntnissen von ROBLES u.a. und DEMPSEY u.a. ist die zweithäufigste Nationalität der Free-Software bzw. Open-Source-Entwickler die deutsche. ROBLES u.a., a.a.O. und DEMPSEY, Bert J.; WEISS, Debra; JONES, Paul; GREENBERG, Jane: A Quantitative Profile of a Community of Open Source Linux Developers. October 1999. Online veröffentlicht unter: http://www.ibiblio.org/osrt/develpro.html. [Aufgerufen am 18.01.2005].

409 MORTON, Andrew: Rede zum Thema „The Policy Implications of Open Source Software". Veranstalter: The Forum on Technology & Innovation. 15.07.2004. Online veröffentlicht unter: http://www.tech-forum.org/upcoming/transcripts/Transcript_OpenSource_07-15-04.pdf [Aufgerufen am 18.12.2004].

Der typische Garagen-Hacker scheint also inzwischen aus der Entwicklung des Linux-Kernels verschwunden zu sein – entgegen den gelegentlich romantisierenden Darstellungen im Internet und in Zeitschriften. Zumindest ein Vorurteil aber läßt sich aufrechterhalten: Nur 1,4 Prozent der Entwickler von freier Software sind weiblich.[410]

3.3 Die Organisation der Projekte

Auch wenn die Linux-Entwickler nicht die ersten waren, die gemeinschaftlich im Netz Software entwickelten, so war ihre Gemeinschaft doch bald die größte. An der Kommunikation über die Linux-Kernel-Mailingliste – dem wichtigsten Medium – beteiligten sich allein von 1995 bis 2000 13.000 Menschen; insgesamt 175.000 Nachrichten wurden in dieser Zeit darüber versandt.[411] Noch heute zählt man dort 400 bis 500 E-Mails am Tag.[412]

Wie diese Zusammenarbeit geregelt wurde, soll im folgenden gezeigt werden. Dabei kann die Organisation der Linux-Kernel-Entwickler als Beispiel für die gesamte Szene dienen – sie wurde aufgrund ihres Bekanntheitsgrades und ihres Erfolgs zu einem Vorbild für viele weitere Entwicklergemeinschaften von freier Software, auch wenn einzelne Ausprägungen verschieden sein können.[413]

3.3.1 Hierarchie

Tanenbaum hatte sich überschüttet gefühlt von den vielen E-Mails zu Minix,[414] Torvalds ging es mit Linux bald ähnlich. Anders als Tanenbaum bestand seine Reaktion aber nicht aus Verweigerung sondern aus der Delegation von Entscheidungen. Diese aus organisationswissenschaftlicher Sicht völlig natürliche strukturelle Entwicklung wurde von ihm unbewußt angestoßen; er selbst kokettiert noch immer gelegentlich damit, daß er es genieße, „die Anerkennung für die Arbeit anderer einzuheimsen“[415].

Die Entwicklung des Linux-Kernels gilt als Paradebeispiel für das Fehlen einer bewußten organisatorischen Gestaltung: „Es gab keine formale organisationale Einheit, kein Businessplan wurde zu Beginn des Projekts entwickelt, kein Budget zugeordnet und kontrolliert, keine Mitarbeiter bewußt ausgewählt und natürlich ist auch keine zentrale räumliche Einrichtung vorhanden gewesen.“[416] Torvalds selbst beschreibt die ungeplante, evolutionäre, um nicht zu sagen: chaotische Weise, auf die die Organisations- und Entscheidungsstruktur bei der Entwicklung des Linux-Kernels entstanden ist:

---

410 ROBLES u.a., a.a.O.

411 Vgl. MOON/SPROULL a.a.O.

412 421 E-Mails waren es am 27.01.2005.

413 So werden etwa die Leiter des Apache-Webserver-Projekts oder der Debian-Distribution von allen beteiligten Entwicklern per Wahl bestimmt. Im Projekt der Programmiersprache Perl rotiert dagegen die Führungsposition innerhalb einer Gruppe von langjährigen Mitarbeitern. Vgl. RAYMOND: Homesteading... a.a.O., S.102.

414 Siehe Fußnote 350.

415 TORVALDS: Just for…, a.a.O., S. 130.

416 BRÜGGE, Bernd; HARHOFF, Dietmar; PICOT, Arnold; CREIGHTON, Oliver; FIEDLER, Marina; HENKEL, Joachim: Open-Source-Software. Eine ökonomische und technische Analyse. Berlin: Springer, 2004, S. 34. BRÜGGE u.a. verweisen in diesem Zusammenhang auf: MOON; SPROULL, a.a.O.

> "I've never made any conscious decisions that this is how I want it to work in the future. All the changes have been automatic responses to different circumstances. At first, people just sent me patches directly, and what happened was I got to trust some people because I'd been e-mailing with them; I knew they were technically sound, and did the right thing (a key hacker concept). Other people started sending patches to these people I trusted because that offloaded some of my work because the people I trusted checked the patches. It's worked pretty well. In some cases I tell people who e-mail me directly, 'Can you send it to this other person just to check?' because I don't have the hardware, for example. The other person's the guy in charge of that part. And then after that, it kind of automatically goes the right way. So there's maybe ten or twenty who are in charge of certain parts of the kernel like networking, SCSI drivers subsystem, whatever."[417]

Aus der hier beschriebenen Praxis entstand eine Hierarchie aus Vertrauensleuten, die als „Lieutenants"[418] oder „Maintainer" bezeichnet werden. Diese sind verantwortlich für die Pflege einzelner Teile des Linux-Kernels, wie etwa Gerätetreiber – oder sogar für die Betreuung ganzer Kernelversionen.[419]

Abbildung 21 Hierarchie globalisiert – die drei wichtigsten Linux-Maintainer wohnen in aller Welt: Alan Cox (Swansea, Wales), Andrew Morton (Sydney, Australien) und Marcelo Tosatti (Porto Alegre, Brasilien).[420]

---

[417] MOODY, a.a.O., S. 81. MOODY zitiert Torvalds aus einem nicht näher bezeichneten Interview im Jahr 1996.

[418] Diese Bezeichnung wird heute eher seltener gebraucht. MOODY zufolge wurde sie geprägt von dem Linux-Entwickler Michael K. Johnson, der erklärte, es handle sich bei Torvalds Vertrauensleuten um „a few trusted lieutenants, from whom he will take larger patches and trust those patches. The lieuts more or less own relatively large pieces of the kernel." MOODY, Glyn: The Greatest OS That (N)ever Was. In: Wired Magazine. Ausgabe 5.08, August 1997. Online veröffentlicht unter: http://www.wired.com/wired/5.08/linux_pr.html [Aufgerufen am 18.01.2005].

[419] Um in der Lage zu sein, neue Entwicklungen am System auszuprobieren, wird ab einer gewissen Reife der Linux-Kernel auf dem bisherigen technischen Stand eingefroren, lediglich Sicherheitsupdates werden noch eingespielt. Eine Kopie dieses „stabilen" Kernels dient dann als Basis für einen sogenannten „Entwicklerkernel", an dem neue Möglichkeiten getestet werden können. Ein Beispiel für diese parallele Arbeitsweise ist der stabile Kernel 2.4 und der Entwicklerkernel 2.5, der dann nach ausgiebiger Testphase wiederum zum stabilen Kernel 2.6 ernannt wurde. Weil auch ältere Kernelversionen häufig noch produktiv eingesetzt werden, gibt es für sie spezielle Vertrauensleute, dies sind Alan Cox (2.2 Kernel) Marcelo Tosatti (2.4-Kernel) und Andrew Morton (Maintainer des vorläufigen Entwicklerkernels 2.6-mm).

[420] Das Bild von Alan Cox zeigt ihn am 5. Oktober 2005. Online veröffentlicht in Wikipedia: http://en.wikipedia.org/wiki/Alan_Cox [Aufgerufen am 10. 10.2005].
Das Foto von Andrew Morton wurde veröffentlicht von CAMERON, Nadia: „Morton: The Linux kernel is no place for 'self-expressive fancy'". In: LinuxWorld, 16/07/2003 10:37:46, online veröffentlicht unter: http://www.linuxworld.com.au/pp.php?id=1546887162&taxid=24 [Aufgerufen am 10. 10.2005].
Das Foto von Marcelo Tosatti stammt aus: KERNEL TRAP: Interview mit Marcelo Tosatti, December 22, 2003, 21:05. Online veröffentlicht unter: http://kerneltrap.org/node/1880 [Aufgerufen am 10. 10.2005].

Zum Maintainer ernannt werden Entwickler im Nachhinein, wenn sie sich bereits als Experten in ihrem jeweiligen Arbeitsbereich herauskristallisiert haben. Es gilt der Grundsatz „authority follows responsibility“[421] – damit bildet sich eine Art Meritokratie, ähnlich wie in der vierten Regel der Hacker-Ethik beschrieben.

> “’It’s very simply a case of who’s actually working on something,’ Tweedie says, ‘and who has got a track record of making good choices. And if you are working on something and you’ve credibility in the eyes of other people who are doing that, then other patches tend to come through you. There are people who are generally recognized as being not owners but simply experts in particular areas. The community takes advantage of that and works through those people to get work done in those areas.’ In other words, ‘it’s very, very much a meritocracy,’ he says.”[422]

Die Namen, Aufgaben und E-Mail-Adressen der Maintainer werden seit Anfang 1996 schriftlich in einer eigenen Datei festgelegt, die zusammen mit dem Quellcode von Linux verteilt wird.[423] Bei dem derzeit aktuellen Kernel 2.6.10 sind 288 Einzelpersonen für 430 Aufgabenbereiche zuständig.[424] 1996 gab es nur drei Maintainer, wie Alan Cox damals mitteilte: Er selbst und Jon Naylor waren für die Netzwerkfähigkeiten von Linux verantwortlich; vom dritten Maintainer wurde keine E-Mail-Adresse genannt, sondern nur eine kurze Bemerkung, die auch als Begründung für diese Art der Organisation dienen kann:

> “REST:
> P: Linus Torvalds
> S: Buried alive in email”[425]

3.3.2 Management

Der engste Kreis der Vertrauensleute trifft sich mindestens einmal im Jahr persönlich mit Torvalds, um die Entwicklungsrichtung zu besprechen.[426] Dabei behält dieser die Entscheidungsgewalt, wie Andrew Morton berichtet:

> “Linus sets a philosophical direction about how he likes the code to be. The rest of us pretty much follows his lead.”[427]

---

[421] RAYMOND: Homesteading the…, a.a.O., S. 102. RAYMOND sieht dies explizit als Hacker-Grundsatz, der bei der Organisation von Open-Source-Projekten angewandt wird.

[422] MOODY, a.a.O., S. 84. MOODY zitiert aus einem eigenen Gespräch mit dem heutigen Maintainer des Dateisystems ext3, Stephen Tweedie.

[423] Die Datei nennt sich „MAINTAINERS“ und wurde zum ersten Mal durch den Kernel-Patch 1.3.68 vom 22.02.1996 in dem damit behandelten Linux-System erzeugt. Urheber war Alan COX, der Titel der Datei lautete: „Maintainers And Source Submission Procedures“. Der Patch ist online verfügbar unter anderem bei: http://sunsite.unc.edu/pub/Linux/kernel/v1.3/patch-1.3.68.gz [Aufgerufen am 18.01.2005].

[424] Hinzu kommen vier Projekte und Firmen, wie etwa Silicon Graphics Inc., die für das von ihnen als Freie Software veröffentlichte XFS-Dateisystem als Maintainer fungieren. Einzelne Maintainer sind für bis zu 13 Aufgabenbereiche verantwortlich.

[425] COX, a.a.O. „P:“ steht für „Person“, „S:“ für „Status“.

[426] BRÜGGE u.a., a.a.O., S. 35.

[427] RIVLIN, Gary: Leader of the Free World. How Linus Torvalds became benevolent dictator of Planet Linux, the biggest collaborative project in history. In: Wired Magazine. Ausgabe 11.11, November 2003, S.157. Online veröffentlicht unter:
http://www.wired.com/wired/archive/11.11/linus_pr.html [Aufgerufen am 18.01.2005].

Innerhalb des Kreises seiner „Lieutenants“ nimmt Torvalds eine dominante Rolle ein. Raymond beschreibt ihn als „benevolent dictator“[428]: Wohlwollend müsse der Führer eines solchen Projekts schon deswegen sein, weil er es nicht als alleiniges Eigentum betrachten könne – er sei gezwungen, anderen die Reputation zukommen zu lassen, die sie sich durch ihre Arbeit an Teilen des Systems erworben haben.[429] Seine Art des Umgangs mit den Kollegen ist daher meistens weder gebieterisch noch diktatorisch, sondern eher von Understatement geprägt:[430] So hieß es schon in seiner erste Mail an das Minix-Forum: „[...] just a hobby, won’t be big and professional like gnu“[431] – diese Zurückhaltung behielt er bei, wie sein technischer Hinweis zur Integration des PCI-Bussystem drei Jahre später zeigt: „Do you feel like you’d actually want to try this out? (hint, hint, Linus wants to get out of doing it himself ;^)”[432]

Der zurückhaltende Kommunikationsstil von Torvalds gilt als eine wichtige Bedingung für den Erfolg des Linux-Projekts,[433] es wäre jedoch falsch, aus seinem eher leisen Tonfall auf einen Mangel an Autorität zu schließen.

> “He never orders anyone to do anything and even his suggestions are mild-mannered. [...] Yet, there is no confusion about who has decision authority in this project. Torvalds manages and announces all releases – all 569 of them to date. He acts as a filter on all patches and new features – rejecting, accepting, or revising as he chooses. He can single-handedly decide to redo something completely if he wishes. [...] As Torvalds says, ‘[There is] one person who everybody agrees is in charge (me)’”[434]

MOON/SPROULL sprechen von Torvalds als „great person“[435], dessen alleinige Führerschaft zwar nicht unbedingt notwendig sei, aber doch wichtige Vorteile bringe: Eine klare Zuordnung der Entscheidungsgewalt, einheitliche Vorgaben und eine in sich konsistente Darstellung des Projekts.[436]

Ob eine solche, an Diktaturen, das Militär oder traditionelle Familienunternehmen erinnernde Führung bei der verteilten und offenen Organisation der Linux-Kernel-Entwicklung tatsächlich möglich ist, darf aber bezweifelt werden. So liegen den Entscheidungen Torvalds für oder gegen die Veränderung des Kernels zwar feste Kriterien zugrunde wie die Qualität des Codes, ein langfristig möglichst geringer Pflegeaufwand oder der Bedarf einer möglichst hohen Anzahl von Nutzern[437] – diese Entscheidungen fallen aber nicht vorab als „Vision“, sondern erst im Nachhinein, wenn ihm die entsprechen-

---

428 RAYMOND, Eric Steven: Homesteading the…, a.a.O., S. 101.

429 Ebd.

430 Vgl. MOON; SPROULL, a.a.O.

431 TORVALDS: What would…, a.a.O.

432 TORVALDS, Linus Benedict: Re: Shared Interrupts – PCI. E-Mail an die Newsgroup „linux.kernel“ vom 29.06.1995 21:03:03 Uhr. Online archiviert unter: http://www.uwsg.iu.edu/hypermail/linux/kernel/9506/0194.html [Aufgerufen am 24.01.2005].

433 Vgl. MOODY, a.a.O., S. 119.

434 MOON; SPROULL, a.a.O. MOON/SPROULL verweisen zur Quelle der letzteren Äußerung von Torvalds auf: YAMAGATA, Hiroo: The Pragmatist of Free Software. Linus Torvalds Interview. In: Wired Japan, 20.09.1997. Online veröffentlicht unter: http://kde.sw.com.sg/food/linus.html [Aufgerufen am 21.01.2005].

435 Ebd.

436 MOON/SPROULL sprechen von „clear locus of decision-making, singular vision, consistent voice“ a.a.O.

437 BRÜGGE u.a., S. 36. BRÜGGE u.a. zitieren als Beleg eine Äußerung von Dan FRYE, Direktor von IBM's Linux Technologie-Zentrum.

den Änderungen (Patches) vorgelegt werden. Torvalds fungiert also nicht als „Führer“, sondern als „Gatekeeper“.[438]

Wenn Torvalds selbst erklärt, daß er grundsätzlich „keine Entscheidungen treffe und den Dingen ihren Lauf lasse“,[439] darf dies nicht als Entscheidungsunlust verstanden werden (oder als Koketterie): Es handelt sich lediglich um eine Absage an ***Vor***entscheidungen und Visionen; Entscheidungen über die Annahme von Patches trifft er jederzeit. Für Organisationsforscher bleibt deswegen die Frage spannend, von wem denn strategische Entscheidungen und Initiativen ausgehen, wenn Torvalds als unbestrittener Anführer vor allem als Gatekeeper agiert, d.h. weitgehend passiv bleibt. Die Antwort liegt auf der Hand: Von den Anwendern und Entwicklern des Systems selbst:

> “There are lots of individuals with some generic feeling about where they want to take the system (and I’m obviously one of them), but in the end we’re all a bunch of people with not very good vision. And that is GOOD. A strong vision and a sure hand sound like good things on paper. It’s just that I have never *ever* met a technical person (including me) whom I would trust to know what is really the right thing to do in the long run. [...] I’d much rather have ‘brownian motion’, where a lot of microscopic directed improvements end up pushing the system slowly in a direction that none of the individual developers really had the vision to see on their own. [...] if you walk in all directions at once, and kind of feel your way around, you may not get to the point you *thought* you wanted, but you never make really bad mistakes, because you always ended up having to satisfy a lot of *different* opinions. You get a more balanced system”[440]

Diese Vorgehensweise, die Entdeckung der Strategie in der Retrospektive, hat der Organisationsforscher WEICK bereits 1979 mit den Worten beschrieben: „How can I know what I think until I see what I say?“[441] IANNACCI sieht in dieser nachträglichen Sinnstiftung das grundsätzliche Management-Prinzip der Linux-Entwicklung; sie folge der Frage: Wie kann ich wissen, was ich denke, bis ich sehe, was *sie* (die anderen Entwickler) tun?[442] Er spricht von einem Managementmodell, bei dem zuerst gehandelt und dann gedacht wird, bei dem die Bedeutung erst erkennbar wird, wenn die Fakten schon geschaffen sind.[443]

---

[438] RAYMOND: The Cathedral…, a.a.O., S. 53.

[439] TORVALDS: Just for…, a.a.O., S. 181.

[440] TORVALDS, Linus Benedict: Re: Coding style – a non-issue. Mail an die Newsgroup „mlist.linux.kernel“ vom 30.11.2001, 22:00:26 PST. Online archiviert unter: http://groups.google.de/groups?q=g:thl3584349692d&dq=&hl=de&lr=&selm=linux.kernel.Pine.LNX.4.33.0111302048200.1459-100000%40penguin.transmeta.com&rnum=83 [Aufgerufen am 25.01.2005].

[441] WEICK, K. E.: The Social Psychology of Organizing. Second edition. Reading, Mass.: Addison-Wesley, 1979. Zitiert nach: IANNACCI, Federico: The Linux managing model. In: First Monday, Ausgabe 8, Nr. 12. Dezember 2003. Online veröffentlicht unter: http://firstmonday.org/issues/issue8_12/iannacci/index.html [Aufgerufen am 18.01.2005].
Das Originalzitat stammt von dem britischen Schriftsteller E. M. Forster – er beschrieb die Reaktion einer alte Dame, nachdem ihr ihre Nichten das Konzept von ‘Logik’ erklärt hatten: „Logic! Good gracious! What rubbish!’ she exclaimed. ‘How can I tell what I think till I see what I say?’ Her nieces, educated young women, thought that she was passée; she was really more up-to-date than they were.“ FORSTER, E.M.: Aspects of the Novel. London: Edwin Arnold, 1927.

[442] IANNACCI, a.a.O.

[443] Auch bei dieser Arbeitsweise der Linux-Community sind Parallelen zur Hacker-Ethik erkennbar, vor allem zur dritten Regel „Mistrust authority“.

3.3.3 Machtbegrenzung

Für Gatekeeper gibt es Grenzen – nämlich dann, wenn zu viele Menschen Einlaß begehren. Im September 1998 zeigte Torvalds zum ersten Mal deutliche Anzeichen von Überforderung. Diese unter dem Titel „Linus doesn't scale“[444] (Linus skaliert nicht) bekanntgewordene Episode begann aus Frust über die Langsamkeit, mit der Torvalds viele eingesandte Patches und Fehlerkorrekturen abarbeitete – obwohl er ihre Aufnahme in den offiziellen Kernel zum Teil bereits seit langem zugesagt hatte bzw. positiven Bescheid von seinen Maintainern hatte. Es stellte sich heraus, daß er aus Zeitmangel viele Einsendungen gar nicht zur Kenntnis nahm und die entsprechenden Mails einfach ungelesen gelöscht hatte.

Unter den Linux-Entwicklern, die darauf mit Unverständnis reagierten, war der amerikanische Student David Miller. Um die Prozedur des Einpflegens von Veränderungen am Kernel-Quellcode für alle einfacher zu machen, installierte er auf einem Server seiner Universität das sogenannte „Concurrent Versions System“ (CVS). Damit konnte zeitgleich an vielen Enden des Codes gearbeitet werden.[445] Allerdings hatte das System einen Nachteil: Einmal eingepflegte Patches ließen sich nicht mehr ohne weiteres zurücknehmen. Als mehr und mehr Entwickler ihre Software in Millers CVS einspeisten, führte das zwar zu einem insgesamt fortschrittlicheren System als dem offiziellen Kernel – aber Torvalds weigerte sich, dieses en bloc zu übernehmen, da er die Änderungen nicht mehr einzeln nachvollziehen konnte.

Er forderte statt dessen die Entwickler auf, mehr Geduld mit ihm zu haben und ihre Patches einfach so lange wiederholt einzusenden, bis er sie zur Kenntnis nahm. Dabei ist sein Tonfall alles andere als „benevolent“:

> “The basic point is that I get a *lot* of patches, and I have to prioritize my work. [...] Quite frankly, this particular discussion (and others before it) has just made me irritable, and is ADDING pressure. Instead, I'd suggest that if you have a complaint about how I handle patches, you think about what I end up having to deal with for five minutes. Go away, people. Or at least don't Cc me any more. I'm not interested, I'm taking a vacation, and I don't want to hear about it any more. In short, get the hell out of my mailbox. Linus.“[446]

Wie damals vielen klar wurde, drohte die Gefahr einer Spaltung des Linux-Projekts. Dieses auch „*Forking*“ genannte Phänomen bedeutet nicht anderes als das, was Tanenbaum Torvalds bereits 1992 prophezeit hatte: Durch die GNU GPL konnte ihm jeder Linux wegnehmen und es selbst weiterentwickeln – solange genügend Anwender und Entwickler mitmachten. Das schien zu diesem Zeitpunkt möglich. Es gelang jedoch, den Konflikt

---

[444] Vgl. MOODY, a.a.O., S. 178. Der Vorwurf „Linus doesn't scale“ tauchte zum ersten Mal auf in: McVOY, Larry: A solution for growing pains. E-Mail an die Newsgroup „muc.lists.linux-kernel“ vom 30.09.1998 13:27:41 Uhr. Archiviert unter: http://list-archive.xemacs.org/xemacs-beta/199809/msg00589.html [Aufgerufen am 25.01.2005].
"Linus skaliert nicht“ spielt auf den damals heiß diskutierten Vorwurf „Linux skaliert nicht“ an, bei dem es um die mangelhafte Leistungsfähigkeit der Software unter Multiprozessorsystemen ging. Vgl Moody, ebd.

[445] Vgl. GRASSMUCK, a.a.O., S. 241f.

[446] TORVALDS, Linus Benedict: Re: 2.1.123 and fbcon.c. E-Mail an die Newsgroup „muc.lists.linux-kernel“ vom 29.09.1998, 23:11:48 Uhr. Online archiviert unter: http://www.ussg.iu.edu/hypermail/linux/kernel/9809.3/0850.html [Aufgerufen am 25.01.2005].

beizulegen – unter anderem durch Vermittlung des selbsternannten „Anthropologen“[447] und „Ministers für Propaganda“[448] der Linux-Community, Eric S. Raymond. Die Kontrahenten Miller und Torvalds trafen sich persönlich. Ebenfalls anwesend war der Programmierer Larry McVoy: Er stellte schließlich mit seinem Programm BitKeeper (BK)[449] eine für Torvalds akzeptable Alternative zu CVS in Aussicht und half so, den Streit beizulegen. Die Überzeugung dieser führenden Linux-Entwickler, daß ein soziales Problem technisch lösbar ist, ähnelt dabei dem technischen Optimismus der frühen Hacker.[450]

Im Jahr 2000 ist zudem durch die Schaffung einer zentralen Organisation zur Weiterentwicklung des GNU/Linux-Systems, des „Open Source Development Labs“ (OSDL) auch erreicht worden, daß wichtige Mitarbeiter wie Torvalds, Morton oder Andrew Tridgell ihre Aufgaben nicht mehr quasi nebenberuflich erbringen müssen – sie sind dort angestellt und arbeiten nun auf Kosten von Sponsoren wie IBM, Hewlett-Packard und Intel am System.

Die durchaus reale Möglichkeit des „Forkings“ macht aber deutlich, daß durch die Rechte, die freie Software wie Linux ihren Nutzern und Programmierern einräumt, die Machtverhältnisse zwischen den einzelnen Projektmitgliedern ausbalanciert werden. Jedem Projektleiter, der nicht in der Lage ist, Entscheidungen im Konsens mit der Mehrheit seiner Entwickler zu treffen, droht das Schicksal, zum Diktator ohne Volk zu werden, und gerierte er sich noch so wohlwollend. Einer ebensolchen sozialen Kontrolle sind auch die Maintainer unterworfen: Bekommt der Projektleiter von ihnen keine Patches mehr, sondern nur noch Beschwerden des „Fußvolks“ über dieselbigen, so wird er auch hier einen Konsens suchen müssen.

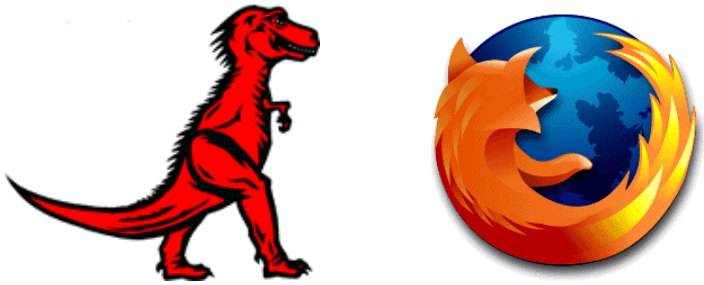


Abbildung 22: Beispiel für eine freundliche Abspaltung: Die Logos der freien Webbrowser-Projekte Mozilla (links) und Firefox.

Kleinere Abspaltungen sind in freien Softwareprojekten durchaus üblich – etwa wenn ein Entwickler eigenwillige Vorstellungen hat und er dafür eine Handvoll Mitstreiter findet. Dies ist kein Problem, sondern belebt die Gemeinschaft – schließlich müssen nach dem

---

447 Vgl. MOODY, a.a.O., S. 178.

448 Vgl. ebd., S. 181.

449 Der wichtigste technische Unterschied zwischen CVS und BK ist, daß ersteres Programm zentral genutzt wird, letzteres aber dezentral. Bei BK bleiben die Patches grundsätzlich nachvollziehbar. Auf Druck von Torvalds hatte sich BK bei der Linux-Entwicklung bis zum Jahr 2005 durchgesetzt, obwohl es im Gegensatz zu CVS keine freie Software ist. Erst nach einer Lizenzänderung im Jahr 2005 ging Torvalds erneut auf die Suche nach einer Alternative zu BK. IANNACI sieht in dem Programm ein ordnendes Prinzip für die Organisation, da es die Entwickler zwinge, zusammenzuarbeiten und ihre Patches gegenseitig zu verifizieren. Vgl. IANNACCI, a.a.O. Eine Gegenüberstellung von CVS und BK findet sich bei: SHAIKH, Maha; CONFORD, Tony: Version Management Tools: CVS to BK in the Linux Kernel. Online verfügbar unter: http://opensource.mit.edu/papers/shaikhcornford.pdf [Aufgerufen am 25.01.2005].

450 Vgl. Kapitel 1.3.6

Copyleft-Prinzip auch die abgespaltenen Entwicklungen frei sein und können so wieder – bei Erfolg – in den Hauptstrang integriert werden. Ein Beispiel für eine solche „freundliche“[451] Abspaltung ist der Webbrowser „Firefox“, der als eine technisch abgespeckte, flinkere Version seines großen Bruders „Mozilla“ begann – sein Erfolg hat inzwischen zu einer Umwandlung des gesamten Projekts geführt.

Neben diesem „freundlichen Fork“ gibt es jedoch auch unfreundlichere Art der Abspaltung. Dabei stirbt der bisherige Entwicklungsstamm ganz ab. Aus seinen Überresten entsteht ein neuer „Setzling“, ein neues Projekt. Dieses Schicksal ereilte Ende 2004 das Xfree86-Projekt, das mehr als zehn Jahre lang die Grundlage der grafische Benutzeroberflächen von GNU/Linux bildete. Der Grund dafür: Der Leiter des Projektes, David Dawes, hatte bei einem Versionswechsel lizenzrechtliche Veränderungen vorgenommen, die nicht mit den Vorgaben für Freie und Open-Source-Software vereinbar waren. Das verärgerte viele seiner Entwickler; die wichtigsten von ihnen schlossen sich zu einem neuen Projekt namens „X.org“ zusammen, wobei ihnen die letzte „freie“ Version von XFree86 als Grundlage diente. Da auch große GNU/Linux-Distributoren wie Red Hat, Suse, Mandrake und Debian ihre Unterstützung für X.org ankündigten, ist das Projekt „Xfree86“ inzwischen weitgehend Geschichte.[452]

Allerdings ist eine solche destruktive Art des Forkings selten. Der ursprüngliche Gründer oder Projektleiter weiß schließlich um die Freiheit seiner Entwickler und wird frühzeitig eine Lösung suchen. Gleichzeitig ist es für einen Abspaltungswilligen schwierig, eine Mehrheit von „Meuterern“ unter seinen Kollegen zu finden, falls er ihnen keine gravierenden Gründe dafür nennen kann. Denn innerhalb der Gemeinschaft verhindern soziale Normen, daß ein Projektleiter einfach so „enterbt“ wird. Diese Normen sollen im folgenden Kapitel untersucht werden.

3.4 Die Koordination der Projekte

Schon herkömmliche Organisationen koordinieren das Verhalten ihrer Mitglieder nicht allein durch Weisungen von Vorgesetzten. Das ist in den Entwicklergemeinschaften von freier Software noch viel weniger der Fall: Freiwilligen Mitarbeitern kann man nichts befehlen. Projektleiter und Maintainer können nur „Anregungen“ aussprechen und Vorstellungen äußern. Bereits im vergangenen Kapitel wurde deutlich, daß die Macht des Projektsleiters lediglich darin besteht, daß er als einziger das Recht hat, neue offizielle Versionen der Software herauszugeben;[453] seine Sanktionsmöglichkeiten beschränken sich auf die Annahme oder Ablehnung von Leistungen.

Häufiger als Weisungen werden in solchen Gemeinschaften die sogenannten Koordinationsmethoden „Programme“ und „Pläne“ angewandt, wobei „Programme“ hier schriftlich fixierte Lösungswege bezeichnen.[454] Als solche Programme sind wohl die „*Frequent-*

---

[451] RAYMOND bezeichnet das als „pseudo-forking; als „forking“ sieht er ausschließlich solche Spaltungen, bei denen der frühere Zweig keine Bedeutung mehr hat. Vgl. RAYMOND: Homesteading the…, a.a.O., S. 72.

[452] Vgl. dazu den Briefwechsel beginnend mit: STALLMAN, Richard Matthew: GPL-incompatible license. E-Mail an die Mailingliste forum@XFree86.Org vom 07.02.2004, 04:01:10 Uhr. Online archiviert unter: http://www.xfree86.org/pipermail/forum/2004-February/003974.html [Aufgerufen am 25.01.2005].

[453] Vgl. RAYMOND: Homesteading the..., a.a.O., S. 73.

[454] Vgl. KIESER; KUBICEK, a.a.O., S. 110-117.

ly Asked Questions“ der verschiedenen Mailinglisten, Newsgroups und Projekt-Homepages anzusehen; darin werden die Ziele erläutert, sowie Konventionen und technische Fragen, auf die sich die Mehrheit der Entwickler geeinigt hat. Zu Programmen sind möglicherweise auch die „*Howto*“-Sammlungen des „Linux Documentation Projects“[455] zu zählen. Auch Pläne werden verwendet, so veröffentlichen die meisten Projekte sogenannte „*Roadmaps*“ zur zukünftigen Entwicklung ihrer jeweiligen Software; allerdings ist auch hier zu berücksichtigen, daß freiwillige Mitarbeiter nicht zu deren Einhaltung zu verpflichten sind.[456]

Hauptsächlich aber kommt von den herkömmlichen Koordinationsmaßnahmen in Arbeitsorganisationen die *Selbstabstimmung* der Mitarbeiter untereinander zur Anwendung.[457] Allerdings kann der Erfolg dieser Koordinationsmethode nicht aus sich selbst heraus erklärt werden: Daß man untereinander auf Augenhöhe verhandelt und sich abstimmt, erklärt noch nicht, warum man in der Lage ist, sich zu einigen.

Das ist nur durch gemeinsame Werte und gemeinsame Motive zu erklären. Erstere versuchen viele traditionelle Unternehmen durch das Mittel der *Organisationskultur* zu stimulieren, also z.B. durch die schriftliche Fixierung einer Unternehmungsphilosophie, durch Rituale und Traditionen, durch gemeinsame Mythen und Helden. KIESER/ KUBICEK nennen das „Management durch Ideologie“.[458]

Das gemeinsame Motiv der Mitarbeiter läßt sich in den meisten Unternehmen in vier Buchstaben ausdrücken: Geld. In freien Softwareprojekten spielt dagegen Geld als Motivation und als Koordinationsmechanismus eine eher untergeordnete Rolle. Im folgenden soll dargestellt werden, welche Motive dort bei Einigungsprozessen und beim Engagement eine Rolle spielen. Anschließend soll ein Blick auf das ideologische Repertoire, auf die Organisationskultur dieser Gemeinschaften geworfen werden.

---

[455] Ein „Howto“ ist eine Anleitung zur Lösung einer bestimmten technischen Aufgabe. Howtos und Software-Dokumentation werden, soweit sie einer freien Lizenz unterliegen, vom „Linux Documentation Project“ veröffentlicht unter http://www.tldp.org/.

[456] Ein Projekt, das sich deswegen grundsätzlich terminlicher Aussagen enthält, ist die Distribution Debian. Auf der Projekt-Homepage heißt es dazu nur: „Wie üblich werden die Release-Ziele und das Release-Datum nicht im Vorhinein bestimmt. Um es einfach zu sagen, ‘Debian veröffentlicht, wenn die Zeit gekommen ist'.“ In: Debian ‘Sarge’ Release-Informationen. http://www.debian.org/releases/sarge/ [Aufgerufen am 25.01.2005]. Zwar gibt es immer wieder Kritik an dieser Praxis; das Fehlen einer Deadline führt aber andererseits dazu, daß Debian als besonders ausgereift gilt: Es wird deswegen allein auf Webservern 468.000mal eingesetzt. Vgl. Netcraft: Slight Linux Market Share Loss for Red Hat. Veröffentlichung des Unternehmens: http://news.netcraft.com/archives/2004/07/12/slight_linux_market_share_loss_for_red_hat.html [Aufgerufen am 25.01.2005].

[457] Vgl. KIESER; KUBICEK, a.a.O. S. 106f. Der für diese Koordinationsweise erforderliche Zeitbedarf wird aufgrund des Fehlens bzw. der Unverbindlichkeit zeitlicher Pläne ausgeglichen. Es ist interessant, daß die Selbstabstimmung, die innerhalb herkömmlicher Arbeitsorganisationen als Mittel zur Koordination „unter Gleichen“ vor allem in der Unternehmensführung eine Rolle spielt, offensichtlich auch unter Freaks und Hackern effektiv funktioniert.

[458] KIESER; KUBICEK, a.a.O., S. 123.

3.4.1 Meritokratie als Entlohnungsform

Obwohl die GPL und andere freie Lizenzen das sogenannte „Forking“ ausdrücklich erlauben, kommt es selten vor. Dieses Phänomen nahm einer der Vordenker der Szene,[459] der amerikanische Neokonservative Eric S. Raymond zum Anlaß, in seinem Essay „Homesteading the Noosphere“[460] die Normen und Werte der GNU/Linux-Entwickler zu analysieren. Er listet dazu mehrere ungeschriebene Verhaltensregeln in freien Softwareprojekten auf, darunter die folgenden:

- Die Spaltung eines Projekts geschieht nur als letzter Ausweg, unter öffentlichen Erklärungen und Rechtfertigungen und erfordert eine Neubenennung,[461]
- Das Verbreiten von inoffiziellen Versionen der Projektsoftware ohne Billigung des Leiters stößt auf allgemeine Ablehnung,[462]
- Das Entfernen von Namen aus Dateien des Projekts wie z.B. der Projektgeschichte („Revision history“), der Liste der Danksagungen („Credits“) oder der Liste der Zuständigkeiten („Maintainers“) ohne Zustimmung der betreffenden Person ist tabu.[463]
- Angriffe auf die Kompetenz anderer Entwickler sind nicht gern gesehen: Wenn jemand einen Fehler findet, ist dies ein Fehler im Programm, kein Fehler eines Programmierers. Anders als im Wissenschaftsbetrieb bringen öffentliche Zweifel an der Kompetenz eines anderen keinen eigenen Nutzen.[464]
- Prahlerei und das zur Schaustellen von persönlichem Ehrgeiz sind verpönt.[465]

Raymond führt diese Verhaltensregeln darauf zurück, daß es sich bei den Entwicklern von freier Software um Hacker-Gemeinschaften handelt, deren Werte auf einer „Geschenkkultur“ basieren – im Gegensatz zu Kulturen, die auf dem Austausch von Waren beruhen oder auf Befehl und Gehorsam.[466]

Er begründet das damit, daß die „Open-Source-Hacker“[467] innerhalb ihrer virtuellen Gemeinschaft weder um physische Werte wie Speicherkapazitäten, Netzwerkbandbreite oder Prozessorgeschwindigkeiten konkurrierten (die seien schließlich Zugangsvoraussetzung), noch um Software (denn die sei schließlich frei und werde geteilt).[468] Der einzige

---

459 Der allerdings nicht unumstritten ist. Vgl. WIKIPEDIA (engl.): Eric S. Raymond. In: http://en.wikipedia.org/wiki/Eric_S._Raymond [Aufgerufen am 25.01.2005].

460 RAYMOND: Homesteading..., a.a.O. Das Wort „Noosphere“ geht zurück auf das griechische Wort „noos“ (Geist); der Begriff findet sich RAYMOND zufolge auch im Werk des Philosophen Teilhard de Chardin. Die „Noosphere“ habe nichts mit dem von William Gibson geprägten Begriff „Cyberspace“ zu tun; während letzterer den Datenraum (bestehend aus Netzwerkverbindungen und Speicherplatz) bezeichne, umfasse die „Noosphere“ seinem Verständnis nach den Raum aller möglichen Gedanken. Das Wort „homesteading“ bedeutet soviel wie „Landnahme“ und wird von Raymond unter Verweis auf John Locke als ursprüngliche Form des Grunderwerbs gesehen: der Inbesitznahme von bisher unbeanspruchtem Gebiet (was natürlich vor dem Hintergrund der amerikanischen Eroberung des Westens verstanden werden muß, möglicherweise aber auch im Kontext der unter Geeks beliebten Fernsehserie „Star Trek": RAYMOND ist Science Fiction Fan). Vgl. ebd., S. 76-79.

461 Vgl. ebd., S. 73.

462 Vgl. ebd.

463 Vgl. ebd.

464 Vgl. ebd., S. 90.

465 Vgl. ebd., S.87-92.

466 Vgl. ebd., S. 80ff.

467 Ebd., S. 81.

468 Vgl. ebd.

Besitz, der in dieser Gemeinschaft erzielt werden könne, sei der Besitz des Vertrauens und der Anerkennung durch andere. Da jeder Mensch natürlicherweise nach einer Verbesserung seines Status strebe,[469] finde innerhalb der Gemeinschaft nun ein Wettbewerb um Reputation statt.[470] Sie könne ausschließlich durch das Verschenken von Leistungen für das Projekt erworben werden.[471] Vor diesem Hintergrund deutet Raymond die oben angeführten „Normen“:

- Der Widerstand gegen das „Forking“ besteht, weil es die bisherigen Teilnehmer des Projekts dem Risiko aussetzt, ihre bis dahin erreichte Reputation zu verlieren.[472]
- Verbreitet jemand eine inoffizielle Version, so verlieren die Projektteilnehmer die Kontrolle über die von ihnen geschriebene Software – für deren Qualität stehen sie aber mit ihrem Ruf.[473]
- In einer Gemeinschaft, die um Reputation konkurriert, entspricht das Löschen von Namen aus der „Credits“-Datei einem Diebstahl – und letztlich einer Schädigung aller, denn sie zerstört das Vertrauen in den Bestand der erarbeiteten Werte.
- Greift man einen anderen Programmierer an, weil dessen Software Fehler enthält, so steht die damit angerichtete Rufschädigung (sein Ruf ist alles, was er zu verlieren hat) in keinem Verhältnis zu seinem Fehlverhalten (wer schreibt schon fehlerfreien Code) – ein solcher Angriff untergräbt außerdem die Bereitschaft aller, Code beizusteuern.
- Prahlerei und offen zur Schau gestellter Ehrgeiz sind kontraproduktiv, weil sie den Verdacht wecken, daß man anderen die ihnen zustehende Reputation verweigern wird.[474]

Raymond entwirft ein ähnliches Bild von einer Meritokratie, wie es auch den frühen Hackern als Ideal vorschwebte: Reputation wird ohne Ansehen der Person nur für die jeweiligen Beiträge vergeben, er sieht diese Beiträge sogar als das einzige verläßliche Kommunikationsmittel:[475]

---

469 Vgl. ebd., S. 80. HOLTGREWE spricht von einem „gut amerikanistisch-individualistischem“ Gesellschaftsmodell RAYMONDs. HOLTGREWE, Ursula: Heterogene Ingenieure: Open Source und Freie Software zwischen technischer und sozialer Innovation. In: GEHRING; LUTTERBECK (Hrsg.), a.a.O., S. 339-352, hier S. 344.

470 Vgl. RAYMOND: Homesteading..., a.a.O., S. 81.

471 Vgl. ebd. S. 85. Dabei ist allerdings einzuschränken, daß dieses Geschenk sich nur für die Gemeinschaft als solches darstellen muß: Wie bereits an anderer Stelle beschrieben, werden zahlreiche Entwickler des GNU/Linux-Systems inzwischen für ihre Tätigkeit bezahlt. Die Behauptung, daß das Verschenken von Leistungen die einzige Methode zum Erwerb von Anerkennung sei, muß m.E. hinterfragt werden: Genauso wie die Reputation eines GNU/Linux-Entwicklers sich förderlich auf dessen Karriere außerhalb der Gemeinschaft auswirken kann, so hat auch die in kommerziellen Projekten erworbene Reputation eines Entwicklers Einfluß auf seine Bewertung innerhalb der Gemeinschaft. Stellte Apple-Chef Steve Jobs eine Frage in der linux.kernel-Mailingliste, erhielte er andere Reaktionen als ein weniger prominenter „Newbie“ (Neuling).

472 Vgl. ebd.

473 Vgl. ebd.

474 Vgl. ebd. S. 91.

475 HOLTGREWE sieht in dem dadurch ausgedrücktem Ideal der vollständigen Separation von Arbeitsrolle und Individuum eine Parallele zu Max WEBERs Modell der bürokratischen Organisation. Vgl. HOLTGREWE, a.a.O., S. 344.

> "In the hacker community, by contrast, one's work is one's statement. [...] there's a strong ethos that quality should (indeed *must*) be left to speak for itself. The best brag is code that 'just works.'"[476]

Genauso wie in der physischen Welt die Werte von Geld oder Naturalien durch den jeweiligen Handelspartner und langfristig durch Konvention festgelegt werden, so existieren auch unter den GNU/Linux-Hackern dauerhafte Vorstellungen darüber, wie viel Anerkennung welcher Programmierleistung zu zollen ist.[477] Raymond versucht, diese Vorstellungen aus seiner Beobachtung in eine allgemeine Form zu überführen und sie so einer Interpretation zugänglich zu machen:

- Wertlos sind Programme, die nicht so funktionieren wie versprochen. Unfertige und fehlerhafte Software muß als solche gekennzeichnet sein.[478]
- Ein Programm mit neuartigen Funktionen hat mehr Wert als eine Software, die lediglich eine Variation des Vorhandenen darstellt.[479]
- Die Veröffentlichung des Programms in wichtigen GNU/Linux-Distributionen[480] gilt als Qualitätsnachweis (ähnlich wie die Publikation eines wissenschaftlichen Texts bei einem bestimmten Verlag).
- Ein von vielen Nutzern verwendetes Programm gilt als bedeutender als ein Nischenprodukt.[481]
- Langweilige Arbeiten beizutragen – etwa Fehlerkorrekturen und Dokumentation – gilt als ehrenvoll.[482]

Die Auflistung[483] macht deutlich, daß eine Leistung dann als wertvoll eingestuft wird, wenn sie Originalität, Können und Zeitaufwand erkennen läßt und einen Bedarf befriedigt. Damit unterscheiden sich die Qualitätskriterien der Hacker-Geschenkkultur aber m. E. nicht wesentlich von denen kommerzieller Softwareprodukte – und auch nicht von anderen Bereichen wie der Wissenschaft oder auch dem Journalismus.

Wie HOLTGREWE richtig entlarvt, liegt der von Raymond beschriebenen „Reputationsökonomie"[484] ein „Modell autarker ProduzentInnen zu Grunde, die sich als Siedlergemeinschaft in den Sphären des Immateriellen die eigenen Werkzeuge selbst herstellen.

---

476 RAYMOND: Homesteading the..., a.a.O., S. 89.

477 RAYMOND hält die Beurteilung des Werts von Code aber für wesentlich komplizierter, als bei anderen Gütern: „It is much harder to objectively distinguish a fine gift from a poor one." Ebd., S. 84.

478 RAYMOND begründet damit die oft sehr niedrigen Versionsnummern von freier Software. Während bei kommerzieller Software die Versionsnummer oft ein Marketinginstrument ist (das suggerieren soll: je höher die Nummer, desto weiter der Fortschritt) bleibt bei freier Software der Versionsstand grundsätzlich so lange unter Eins bis das Programm als fehlerfrei betrachtet wird. Vgl. ebd., S. 94.

479 Vgl. ebd., S. 95.

480 Ein Distributor ist eine Art Zwischenhändler, der sich aus den im Internet verfügbaren freien Programmen die herauspickt, die ihm für seine Zwecke am nützlichsten erscheinen (z.B. die besten Server-Programme, die besten Office-Programme) und sie so kombiniert, daß sie reibungslos zusammenarbeiten. Da die wenigsten Menschen Lust haben, sich ein solches System selbst zusammenzubauen, gibt es eine große Nachfrage nach solchen „Distributionen". Vgl. Kapitel 3.5.

481 Vgl. ebd., S. 95f.

482 Vgl. ebd., S. 96.

483 RAYMOND führt noch ein weiteres Kriterium an ("Non-trivial extensions of function are better than low-level patches and debugging" Ebd.), diese scheint mir aber nur eine Konsequenz aus den zuvor ausgeführten Kriterien und auch zu wenig allgemein zu sein.

484 HOLTGREWE, a.a.O., S. 342.

Hier wird ein Stück weit nostalgisch eine Ganzheitlichkeit des Produzierens beschworen, die quasi handwerklich gedacht ist.“[485] Aber selbst wenn seine Analysen auf einer idealistischen Grundlage beruhen, so müssen sie doch als Ausdruck des Selbstbewußtseins der GNU/Linux-Gemeinschaft gewertet werden.[486]

Reputation wird inzwischen weitgehend als ein wichtiges Mittel zur Koordination[487] bei der Entwicklung von Open Source und Free Software anerkannt.[488] Umfragen zufolge streben allerdings die wenigsten[489] Entwickler von freier Software ausdrücklich danach, was HOLTGREWE als Meßfehler interpretiert: Reputation könne immer nur als Nebenfolge einer Tätigkeit angestrebt werden. Die Befragten würden es daher oft nicht zugeben – oder gar nicht sehen –, daß die eigene Tätigkeit nicht nur der Sache, sondern auch der Befriedigung des Egos diene.[490]

Neben dem Prinzip der Reputation findet man auch das zweite Element einer Meritokratie, nämlich ihre Offenheit nach außen und ihre Diskriminierungsfreiheit, für die Entwicklergemeinschaft von freier Software häufig bestätigt, etwa bei ROBLES et al.:

> “When a developer wants to participate in a Libre Software project, nobody will ask him neither for his nationality, nor for his profession nor for his age. If he wants to contribute in a constructive way and has the necessary knowledge or skills, yet even if he is only just on the way to acquire them, he is invited to do so.”[491]

Bei näherer Betrachtung erscheint dieses Idealbild einer Meritokratie durchaus hinterfragenswert, denkt man etwa an die bereits beschriebene Geschlechterverteilung. HOLTGREWE stellt dazu pointiert fest: „EntwicklerInnen [...] sind mit 98 – 99% fast ausschließlich männlich.“[492] Die meisten Mitarbeiter an GNU/Linux-Projekten sind außerdem unter 30 Jahre alt und 60 bis 70 Prozent von ihnen haben einen Hochschulabschluß[493] – sie sind damit nicht gerade ein repräsentativer Querschnitt ihrer Gesellschaften, nicht einmal der westlichen Industriestaaten,[494] in denen die meisten der Beteiligten wohnen. Ganz offensichtlich ist „[...] die Beteiligung an OS/FS-Projekten bei aller proklamierter Offenheit

---

485 Ebd.

486 Vgl. RAYMOND: Homesteading the..., a.a.O., S. 67.

487 Als solches Instrument wird das Reputations-Modell inzwischen auch in einem kommerziellen Online-Projekt eingesetzt: In Form des Bewertungsverfahrens beim Internet-Auktionshaus Ebay. Vgl. BORCHERS, Detlef: Trau keinem unter www. So wichtig Vertrauen in der Online-Welt ist, so hart muss man es sich erarbeiten. Süddeutsche Zeitung, 12.08.03, S. 17 (Wissenschaft). BORCHERS berichtet dabei über die Tagung „Trust and Community on the Internet. Possibilities and Restrictions for Cooperation on the Internet“, die vom 31. Juli bis zum 2. August 2003 am Zentrum für interdisziplinäre Forschung der Universität Bielefeld stattfand. Eine Dokumentation dieser Tagung findet sich unter: http://www.uni-bielefeld.de/ZIF/AG/2003/07-31-Baurmann.html [Aufgerufen am 08.08.2004].

488 Vgl. BRÜGGE, Bernd; HARHOFF, Dietmar; PICOT, Arnold; CREIGHTON, Oliver; FIEDLER, Marina; HENKEL, Joachim: Open-Source-Software. Eine ökonomische und technische Analyse. Berlin: Springer, 2004, S. 59.

489 Vgl. GHOSH, Rishab A.; GLOTT, Rüdiger; KRIEGER, Bernhard; ROBLES, Gregorio: Free / Libre and Open Source Software: Survey and Study. Deliverable D 18: Final Report, Part 4: Survey of Developers. Online veröffentlicht unter: http://floss1.infonomics.nl/finalreport/FLOSSfinal-4.pdf. [Aufgerufen am 26.01.2005].

490 HOLTGREWE, a.a.O., S. 346.

491 ROBLES u.a., a.a.O., S. 2.

492 HOLTGREWE, a.a.O., S. 344f.

493 Ebd., S. 345.

494 Vgl. ROBLES u.a., a.a.O.

an bestimmte soziale und subjektive Voraussetzungen gebunden, die sich bei bestimmten sozialen Gruppen, jüngeren männlichen Hochqualifizierten, konzentrieren".[495]

HOLTGREWE nennt das die „Ausblendungen"[496] bei der „Beschwörung der virtuellen Pioniergemeinschaften".[497] In der Tat mag man manchen Liebhabern freier Software eine idealisierte Darstellung von Offenheit und Diskriminierungsfreiheit vorwerfen. Allerdings sind die Mitglieder solcher Gemeinschaften nicht dafür verantwortlich zu machen, daß die nötigen Zugangsvoraussetzungen wie Bildung und eine zumindest moderate materielle Freiheit nur von einem Bruchteil der Mitglieder ihrer Herkunftsgesellschaften erreicht werden.

3.4.2 Motivation der Mitarbeiter

Teil einer Gemeinschaft zu sein, in der die eigene Leistung zählt und gerecht bewertet wird, ist für viele eine motivierende Erfahrung, nicht nur für Programmierer. Die Teilnahme an solchen Projekten ist aus dem Streben nach Reputation allein jedoch nicht zu erklären, schließlich beginnt man als Neuling zunächst einmal bei Null. Im folgenden sollen daher die wichtigsten auslösenden Motive für ein Engagement dargestellt werden.

3.4.2.1 Eigennutz

Raymond sieht als ersten Schritt in ein freies Projekt grundsätzlich den eigenen Bedarf: „Every good work of software starts by scratching a developer's personal itch. [...] it's long been proverbial that 'Necessity is the mother of invention'."[498] Diese These finden LAKHANI und WOLF in ihrer Umfrage unter 648 Entwicklern von freier Software bestätigt: 58 Prozent der Befragten bezeichneten dieses Motiv als wichtig.[499] Die Programmierer nehmen dabei entweder den Code des Projekts und passen das Programm an ihren jeweiligen Bedarf an, oder sie haben selbst bereits eine entsprechende Lösung entwickelt, die sie an das Projekt weitergeben möchten bzw. zu der sie Fragen haben. Raymond sieht dies nicht nur als Motiv für den Einstieg in ein bestehendes Projekt an, sondern auch als Ausgangspunkt für den Start eines neuen.[500]

---

[495] HOLTGREWE, a.a.O., S. 344.

[496] Ebd., S. 342.

[497] Ebd.

[498] RAYMOND: The Cathedral…, a.a.O., S. 23.

[499] LAKHANI, Karim R.; WOLF, Robert G.: Why Hackers Do What They Do: Understanding Motivation Effort in Free/Open Source Software Projects. In: FELLER, J.;FITZGERALD, B.; HISSAM, S.; LAKHANI, Karim R. (Hrsg.): Perspectives on Free and Open Source Software. Boston: MIT-Press, 2005. S.11. Nach der Online-Ausgabe unter: http://opensource.mit.edu/papers/lakhaniwolf.pdf [Aufgerufen am 26.01.2005].

[500] RAYMOND: The Cathedral…, a.a.O., S. 23.

3.4.2.2 Lernen

Hohe Zustimmungsraten in mehreren Umfragen erhielt auch das Motiv des Lernens für die Beteiligung an solchen Software-Projekten.[501] Dies wird auch durch die bereits beschriebenen Entwicklungen bestätigt: So hatte Tanenbaum das Betriebssystem Minix geschrieben, um überhaupt seinen Studenten modernen Code von Betriebssystemen zeigen und sie daran experimentieren lassen zu können – dies legal am originären Unix-Code zu tun, war inzwischen zu teuer für die Hochschulen. Auch GRAHAM berichtet, wie er vor der Existenz von freier Software das Programmieren anhand von veralteten Beispielen und Raubkopien hatte lernen müssen.

> "One of the less publicized benefits of the open source movement is that it has made it easier to learn to program. When I learned to program, we had to rely mostly on examples in books. The one big chunk of code available then was Unix, but even this was not open source. Most of the people who read the source read it in illicit photocopies of John Lion's book, which though written in 1977 was not allowed to be published until 1996."[502]

Neben dem Lesen und Lernen am fremden Code ist auch das Ausprobieren eigener Fähigkeiten in einem solchen Projekt möglich – es wurde bereits dargestellt, daß dieses Motiv für Torvalds ausschlaggebend für seine Beschäftigung mit Minix und der Entwicklung eines eigenen Unix-Systems gewesen ist.

BORCHERS zitiert sogar eine finnische Untersuchung, nach der die Mitarbeit in freien Softwareprojekten eine bessere Schule für Programmierer sei als eine Informatiker-Ausbildung.[503] Eine mögliche Begründung dafür liefert HIMANEN: Er sieht in der Gemeinschaft der Entwickler freier Software das Modell der offenen Akademie verwirklicht, dieses sei der herkömmlichen Ausbildung weit voraus:

> „Das offene Lernmodell der Hacker kann man als ihre ‚Netz-Akademie' bezeichnen. Damit meinen wir eine sich kontinuierlich erweiternde Lernumwelt, die von den Hackern selbst geschaffen wird. [...] Wieder einmal erinnert das Hacker-Modell an Platons Akademie, in der Schüler nicht als Objekte der Wissensvermittlung betrachtet, sondern als Partner beim Lernen bezeichnet wurden (*synetheis*).[504] [...] Das Ethos des ursprünglichen akademischen Modells und des Hacker-Modells – treffend in Platons Vorstellung zusammengefaßt, daß ‚kein freier Mensch etwas wie ein Sklave lernen sollte' – unterscheidet sich ganz und gar von dem des Klosters (der Schule), dessen Geist in der Benediktinerregel ausgedrückt ist: ‚Denn Reden und Lehren kommt dem Meister zu, Schweigen und Hören ist Sache des Jüngers.' Ironischerweise neigt die Akademie derzeit dazu, ihre Lernstruktur am klösterlichen Sender-Empfänger-Modell zu orientieren."[505]

---

[501] Vgl. LUTHIGER: Alles aus..., a.a.O., S. 96. Er verweist dabei auf die Studien von GHOSH et al., a.a.O., LAKHANI; WOLF, a.a.O. und HARS, A.; OU, S.: Working for free? Motivations of Participating in Open Source Projects. 34th Annual Hawaii International Conference on System Sciences.

[502] GRAHAM, a.a.O., S. 27.

[503] BORCHERS, Detlef: 'Open Source' – das Zauberwort des Jahres 1998. Wenn ein kleiner Drache Feuer spuckt. Warum hochwertige Software nichts kosten muß und dennoch große Geschäfte verspricht. Süddeutsche Zeitung, 29.12.98. S.10 (Computer).

[504] HIMANEN verweist an dieser Stelle auf: PLATON: Minos, nach der Übersetzung Friedrich Schleiermachers. In: Sämtliche Werke, Bd. X, Frankfurt am Main: 1991, S. 319e.

[505] HIMANEN, a.a.O., S. 98ff.

Aus der persönlichen Erfahrung mit GNU/Linux[506] soll an dieser Stelle kurz der von HIMANEN angeführte partnerschaftliche Umgang zwischen Lehrer/Experte und Schüler/Neuling in der GNU/Linux-Community bestätigt werden. Den Autoren beeindruckte vor allem die Bereitwilligkeit, mit der internationale Entwickler die Fragen eines Newbies (wie ich es damals war) beantworteten, in mehrtägiger E-Mail-Korrespondenz Probleme lösen halfen oder einfach von sich aus auf Verbesserungsmöglichkeiten hinwiesen.[507] Diese Erfahrungen verstärken die ursprüngliche Motivation und führen dazu, daß man später ebenfalls nur zu sehr geneigt ist, anderen Neulingen Zeit und Gehör zu schenken.

3.4.2.3 Altruismus

Das in den Texten von Stallman zitierte Motiv des „Sharing", des Teilens von Kenntnissen und die Ablehnung des „Hoarding", des Hortens von Software ist ebenfalls für viele Teilnehmer an Projekten von freier Software ein Motiv. LUTHIGER sieht es dabei als gleichgültig an, ob dieser Altruismus aus ideologischen Motiven erfolgt (etwa: aus Engagement für die Freiheit) oder aus einem Gefühl der Reziprozität (etwa weil man als Anwender von freier Software profitiert und deswegen das Bedürfnis hat, etwas „zurückzugeben").[508] Nach der Studie von LAKHANI und WOLF bezeichnet ein Drittel der Befragten diese Haltung als relevant für ihr Engagement.[509]

Altruismus ist im übrigen nicht nur ein wichtiges Motiv für die Beteiligung an freien Softwareprojekten, sondern auch die Arbeitsgrundlage der gemeinschaftsbasierten Online-Enzyklopädie Wikipedia.[510] Das Prinzip von Wikis ist „Assume good faith – Setze guten Willen voraus".[511] Da jeder jederzeit alles ändern kann, muß die Haltung derer, die sich beteiligen, grundsätzlich optimistisch sein.[512]

---

506 Die Installation eines Apache-Webservers unter Debian GNU/Linux auf einem 486SX Laptop wurde ebenfalls aus Interesse an der persönlichen Weiterbildung begonnen – und einem gewissen winterlichen Bastlerwahn. Der Verlauf des eigenen Projekts wurde im Web dokumentiert.

507 So kamen E-Mails von John Coffman (Maintainer LILO), David Hinds (PCMCIA) und Bart Samwel (Laptop-Mode). Um die Prominenz zu verdeutlichen: In der Windows-Welt hätte man damit Post von dem Microsoft-Kollegen erhalten, der den Startvorgang mit dem blauen Himmel (und den Wolken) geschrieben hat, oder von einem anderen, der allein dafür verantwortlich ist, daß Windows eine PC-Karte, die man in den Laptop steckt, auch erkennt. Ich benutze seit 13 Jahren Windows – es hat noch nie jemand von Microsoft geschrieben.

508 Vgl. LUTHIGER: Alles aus..., a.a.O., S. 96f.

509 LAKHANI; WOLF, a.a.O.

510 Alle Einträge des Wikipedia-Projekts sind urheberrechtlich durch eine Lizenz des GNU-Projekts geschützt, der „GNU Free Documentation License" (GFDL); diese war ursprünglich dafür gedacht, Software-Handbücher ebenso frei verbreiten zu können, wie die unter GPL stehenden Programme.

511 BAUER, Elisabeth; FROMMEL, Oliver: Auf Freiheit programmiert. Konferenz Wizards of OS. Linux Magazin, Ausgabe 08/04. München: Linux New Media AG, 08.07.2004. S. 83.

512 Selbst Vandalen werden bei Wikipedia auf amüsante Weise integriert: Scherze und Verfälschungen von Artikeln werden zwar im Durchschnitt nach fünf Minuten von anderen Nutzern entdeckt und dort entfernt, aber nicht unbedingt endgültig gelöscht. Die dümmsten und brillantesten „Hacks" werden auf einer eigenen Seite des Projekts archiviert.
http://en.wikipedia.org/wiki/Wikipedia:The_original_bad_jokes_and_other_deleted_nonsense [Aufgerufen am 26.01.2005].

3.4.2.4 Spaß

Am häufigsten für das Engagement in freien Softwareprojekten wird als Motiv schlicht „*Spaß*“ genannt.[513] So betitelte Torvalds sein Buch über die Geschichte der Linux-Entwicklung mit „Just for fun“[514] und schloß auch an anderer Stelle ideologische Gründe für sein Engagement mit aller Deutlichkeit aus (und grenzte sich damit von GNU-Gründer Stallman ab):

> “Quite frankly, I don’t *want* people using Linux for ideological reasons. 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’.”[515]

Als Seitenhieb auf Stallmans bekannte Definition von „Freier Software“ als „’free’ as in ‚free speech,’ not as in ‚free beer’“[516] wird die Spaß-Haltung auch als „In for the beer“[517] bezeichnet.

Abbildung 23: „Gehacktes“ Linux-Maskottchen – Beispiel für den Spaß seiner Programmierer.[518]

Sichtbares Zeichen des „Spaßfaktors“ von Linux ist das Maskottchen, der Pinguin „Tux“[519]. Torvalds Vorgaben für dessen Aussehen machen deutlich, daß hier kein Wirtschaftsunternehmen ein dynamisches Logo in Auftrag gab:

---

[513] Möglicherweise muß vor dem Hintergrund dieses „Spaßprinzips“ RAYMONDS Reputationsmodell auch als Regulativ verstanden werden: Die hohe Attraktivität spaßversprechender Vorhaben wird ausgeglichen durch eine hohe Reputation für das Erledigen langwieriger und mühsamer Arbeiten, wie etwa Dokumentation und Fehlersuche.

[514] TORVALDS: Just for..., a.a.O.

[515] TORVALDS, Linus Benedict: Re: [PATCH] Remove Bitkeeper documentation from Linux tree. E-Mail an die Newsgroup „linux.kernel“ vom 20.04.2002, 11:56:21 Uhr. Online archiviert unter: http://www.uwsg.iu.edu/hypermail/linux/kernel/0204.2/1018.html [Aufgerufen am 26.01.2005].

[516] STALLMAN, Richard Matthew: Free Software Definition, a.a.O., S. 41.

[517] Vgl. FISH, Shlomi: The „In for Free Beer“ Manifesto. Online veröffentlicht unter: http://fc-solve.berlios.de/oss-fs/in_for_free_beer_manifesto.html [Aufgerufen am 26.01.2005]. FISH zitiert als Herkunft des Begriffs ein Interview des Online Magazins „Linux Weekly News“.

[518] Normalerweise erscheint der ganz links abgebildete Pinguin beim Starten des Systems in der linken oberen Ecke des Bildschirms. Durch einen Hack wurde durch mehrere Kernel-Versionen hindurch dieses Bild immer wieder so verändert, daß der Pinguin ein Bierglas (Abbildung Mitte) oder ein Weinglas (rechts) in der Flosse hält. Weil in einigen Ländern dieser Scherz auf Unverständnis stieß, wurden die „politisch unkorrekten“ Logos später als Bug gekennzeichnet und „repariert“. Vgl. UYTTERHOEVEN, Geert. The Linux Frame Buffer Device Penguin Logo. Online veröffentlicht unter: http://members.chello.be/cr26864/Linux/fbdev/logo.html [Aufgerufen am 26.01.2005].
Der Original-Pinguin stammt von Larry Ewing. Siehe: http://www.isc.tamu.edu/~lewing/linux/ [Aufgerufen am 26.01.2005].

[519] Für tuxedo, engl.: Smoking.

> „Sein Pinguin sollte glücklich aussehen, so als hätte er gerade eine Maß Bier genossen und den besten Sex seines Lebens gehabt. Und was noch wichtiger war: Er sollte unverwechselbar sein. Während alle anderen Pinguine einen schwarzen Schnabel und schwarze Füße haben, hat das Linus-Maskottchen deshalb einen orangefarbenen Schnabel und orangefarbene Füße, so daß es wie ein Pinguin aussieht, dessen Mutter eine Ente war. Als hätte Daisy Duck sich auf einer Antarktis-Kreuzfahrt vergessen und einen wilden One-Night-Stand mit einem einheimischen Federvieh gehabt."[520]

Die Ironie, die aus der Wahl dieses Logos ebenso spricht, wie aus mündlichen und schriftlichen Äußerungen von Torvalds,[521] Stallmans[522] und vielen weniger prominenten Entwicklern kann dabei auch als Ausdruck des Selbstbewußtsteins verstanden werden, das sie aus ihrer Arbeit beziehen: In der Umfrage von LAKHANI und WOLF bezeichneten 61 Prozent der Befragten ihre Arbeit an freien Softwareprojekten als die in ihrem Leben bedeutendste Erfahrung eigener Kreativität.[523] 73 Prozent der Befragten geben an, sie verlören beim Programmieren grundsätzlich das Zeitempfinden, was als Hinweis auf das Gefühl von „Flow" gedeutet wird.[524]

Dieses von dem (ungarischstämmigen) amerikanischen Psychologieprofessor Mihaly Csikszentmihalyi beschriebene Phänomen drückt das Erleben eines glatten Handlungsablaufs aus: „Ein Schritt geht flüssig in den nächsten über, als liefe das Geschehen gleitend, wie aus einer inneren Logik. [...] Es kommt zur Ausblendung aller Kognitionen, die nicht unmittelbar auf die jetzige Ausführungsregulation gerichtet sind."[525]

Das Ausmaß, in dem Entwickler von freien Softwareprojekten solcherart in ihrer Arbeit aufgehen, untersucht derzeit die Studie „Fun and Software Development" (FASD) am Lehrstuhl für Unternehmensführung und -politik der Universität Zürich. Erste Ergebnisse[526] deuten auf eine hohe Erfahrung von „Flow" hin.

3.4.3 Identitätsstiftung durch Organisationskultur

Bei Unternehmen bezeichnet der Begriff „Organisationskultur" übereinstimmende Überzeugungen ihrer Mitarbeiter. Durch bestimmte Aktionen und Mechanismen hofft man, diese Überzeugungen zu beeinflussen, denn „[...] wenn die Mitglieder einer Organisation übereinstimmende Werte und Normen ‚verinnerlicht' haben – sich mit ihnen identifizie-

---

[520] TORVALDS, Linus Benedict: Just for…, a.a.O., S. 151.

[521] Als Beispiel für viele eine Bemerkung aus seinem Buch „Just for fun" (am Ende eines Kapitels über Ideologie und GNU/Linux): „Die einzige andere Sache, die Wutausbrüche rechtfertigt, sind Leute mit allzu ausgeprägtem missionarischen Eifer. Es gibt einfach keinen Grund, andere bekehren zu wollen und sich dabei so selbstgerecht zu gebärden. Und ich klinge wie einer von ihnen." Ebd., S. 210.

[522] So beschreibt Stallman in seiner wichtigsten Programmatik, dem „GNU Manifesto", seine Utopie einer Welt ohne proprietäre Software: „People will be free to devote themselves to activities that are fun, such as programming, after spending the necessary ten hours a week on required tasks such as legislation, family counseling, robot repair, and asteroid prospecting". STALLMAN: The GNU Manifesto, a.a.O., S. 39.

[523] Vgl. LAKHANI; WOLF, a.a.O., S. 11.

[524] Vgl. ebd.

[525] LUTHIGER: Alles aus..., a.a.O., S. 100. Zu weiteren Beschäftigung mit Flow verweist Luthiger auf: CSIKSZENTMIHALYI, Mihaly: Flow. The Psychology of Optimal Experience. New York: Harper & Row. 1990.

[526] LUTHIGER, Benno: Fun and Software Development. FASD Study – Provisional Results. Version 1.0 Zürich, July 2004. Online veröffentlicht unter: http://www.isu.unizh.ch/fuehrung/blprojects/FASD/FASD_eval1.pdf [Aufgerufen am 26.01.2005].

ren –, können sie ihre Aktivitäten auch ohne strukturelle Vorgaben aufeinander abstimmen“.[527] Zu diesen Mechanismen zählen etwa die schriftliche Fixierung von Grundsätzen, die Etablierung von Ritualen, Symbolen und Mythen[528] sowie die Stilisierung von Helden.[529]

Anders als in herkömmlichen Unternehmen gibt es in freien Softwareprojekten wie GNU/Linux keine zentrale Instanz, die die Macht hätte, unter den Entwicklern Rituale wie etwa den „Wal-Mart-Chant“[530] durchzusetzen. Dennoch gibt es Indizien für eine funktionierende Organisationskultur: So beobachteten HERTEL et al. unter Linux-Entwicklern eine hohe *Identifikation* mit der Gruppe,[531] was vor allem die zeitliche Dauer ihres Engagements positiv zu beeinflussen scheint.[532]

Die hohe Identifikationswirkung der freien Projekte läßt sich zum Teil sicher durch die bereits beschriebene gemeinschaftliche Art der Arbeit, die faire Bewertung nach Reputation, die partnerschaftliche Form des Dialogs und das hohe Ausmaß an persönlicher Befriedigung durch die Tätigkeit des Programmierens in dieser Umgebung erklären. Allerdings trifft man auf der anderen Seite selbst bei flüchtiger Untersuchung immer wieder auf Symbole und Darstellungen, die ideologisch zu sein scheinen und auf eine Organisationskultur hinweisen. Im folgenden sollen einige dieser Hinweise dargestellt werden.[533]

3.4.3.1 Religion

Der Umgang mit Religion ist innerhalb der GNU/Linux-Szene ambivalent. So distanziert sich Stallman öffentlich davon: In einem seiner programmatischen Texte heißt es: „As an atheist, I don’t follow any religious leaders“[534] und wie WILLIAMS berichtet, macht er das auch im alltäglichen Umgang klar:

---

527 KIESER; KUBICEK, a.a.O., S. 118.

528 Als Mythen bezeichnen KIESER und KUBICEK Geschichten wie die vom schwäbischen Unternehmensgründer Bosch: Der soll einmal eine Büroklammer vom Boden aufgeklaubt und einen Angestellten gefragt haben, was das sei. Als dieser auf eine Büroklammer tippte, antwortete Bosch: „Nein, das ist mein Geld“. Vgl. ebd., S. 121.

529 Diese Maßnahmen reichen von Betriebsausflügen und Willkommensveranstaltungen über Kleiderordnungen, die Einführung (oder Verhinderung) von Statussymbolen (Firmenwagen, -laptop, etc.) bis zu immer wieder weitergetragenen Anekdoten über die Gründer des Unternehmens oder besonders verdiente Mitarbeiter. Vgl. ebd., S. 120ff.

530 Dabei kommen die Mitarbeiter des Kaufhaus-Riesen gemeinsam vor Beginn der Arbeitszeit zusammen, um die Freude an der Arbeit zu besingen, während sie Bewegungen wie den „Squiggle“ ausführen; eine Art Hüftschwung, den Wal-Mart-Gründer Sam Walton selbst erfunden haben soll. Vgl. MINK, Andreas: Schmutzige Kosten von Billigpreisen. Zürich: NZZ am Sonntag. 09.11.2003, 02:27, Online kostenpflichtig abrufbar unter: http://nzz.gbi.de/webcgi?START=A20&T_FORMAT=5&DOKM=709962_NZZ_0&WID=49872-8270855-90662_3 [Aufgerufen am 27.01.2005].

531 HERTEL, G.; NIEDNER, S.; HERRMANN, S.: Motivation of Software Developers in Open Source Projects. Research Policy, 32:7, 2003. S. 1159-1177. Zitiert nach LUTHIGER: Alles aus..., a.a.O., S. 96.

532 Ebd.

533 Schriftliche fixierte Grundsätze ("Unternehmungsphilosophien“) sind in der GNU/Linux-Szene durchaus üblich – kein Wunder in einer Kultur, die fast vollständig auf der Schriftform beruht. Ein Beispiel wurde bereits mit der „Free Software Definition“ in Kapitel 2.3 vorgestellt. Vgl. STALLMAN: Free Software Definition, a.a.O., Eine Darstellung von Teilen der „Open Source Definition“ findet sich in Kapitel 3.5. Vgl. PERENS: The Open…, a.a.O.

534 STALLMAN: The GNU Operating..., a.a.O., S. 56.

> "At one point, my wife uttered an emphatic 'God forbid' only to receive a typical Stallman rebuke.
> 'I hate to break it to you, but there is no God,' Stallman said."[535]

Ungeachtet dessen trat er gelegentlich nach seinen Reden als „St. IGNUcius" auf: Verkleidet im Priestergewand, mit einer CD als Heiligenschein, „exorzierte" er Rechner von proprietärer Software;[536] eine Darbietung, die er als Selbstironie bezeichnet – andere sahen sich dadurch in ihrem Vorurteil bestätigt, Stallman sei ein „Software-Ideologe".[537]

Abbildung 24: Richard Stallman als „St. IGNUcius" mit Heiligenschein und Laptop[538]

Religiöse Vergleiche von Stallman finden sich zuhauf: So sieht ihn etwa LESSIG als Moses: Wie dieser sei er ein unerbittlicher Führer ins gelobte Land freier Betriebssysteme, wie Moses habe er das letzten Wegstück einem anderen (Torvalds) überlassen müssen.[539] Stallman selbst kokettiert mit solchen Darstellungen:

> "Some people do compare me with an Old Testament prophet, and the reason is Old Testament prophets said certain social practices were wrong. They wouldn't compromise on moral issues. They couldn't be bought off, and they were usually treated with contempt."[540]

Zwar können Google-Suchergebnisse kaum wissenschaftlichen Kriterien genügen, doch es ist verblüffend, daß bei einer Abfrage mit den Worten „Guru" und „Stallman" insgesamt immerhin 35.400 Ergebnisse gefunden werden – Stichproben förderten dabei Ergebnisse wie Interviews und Zeitungsartikel zu Tage.

Bei einer eingehenden Untersuchung ideologischer Merkmale der Bewegung müßte Stallman damit wohl als Hauptverdächtiger gelten; auf den ersten Blick aber hätten auch

---

535 WILLIAMS, a.a.O., S. 198.

536 WILLIAMS, a.a.O., S. 116f.

537 Vgl. WILLIAMS, a.a.O., S. 117. WILLIAMS verweist in diesem Zusammenhang auf eine Äußerung von Eric S. RAYMOND in einem Interview auf der Webseite „linux.com" im Jahr 1999.

538 Quelle des Fotos: WILLIAMS, a.a.O., S. 116f. Frei veröffentlicht unter: http://www.oreilly.com/openbook/freedom/ch08.html

539 Vgl. LESSIG: The Future..., a.a.O., S. 270.

540 WILLIAMS, a.a.O., S. 75.

die selbsternannten „pragmatischen“ Anführer keine weiße Weste: So berichtet Torvalds von seiner Erfahrung bei einem Linux-Anwendertreffen der Firma Red Hat:

> „Der Saal war brechend voll. Als ich auf die Bühne kam, erhoben sich alle von ihren Plätzen und jubelten mir zu. Die ersten Worte aus meinem Mund waren das Erstbeste, was mir in den Sinn kam: ‚Ich bin euer Gott.’ *Verdammt noch mal, das sollte ein Witz sein*. Ich wollte damit nicht sagen: ‚Ich bin zutiefst davon überzeugt, daß ich euer Gott bin und daran sollt ihr immer denken.’ Ich wollte damit sagen: ‚Okay, okay, okay. Ich weiß ich bin euer Gott. Jetzt setzt euch bitte einfach hin und wartet mit dem Jubel, bis ihr gehört habt, was ich zu sagen habe, obwohl ich euren Vorschußjubel aufrichtig zu schätzen weiß.’ [...] Ein paar Stunden später waren meine einleitenden Worte das Thema von Newsgroup-Postings geworden.“[541]

Auch Vordenker Raymond, der Torvalds’ Autorität in Newsgroups mit den Worten „Linus is god“[542] zu unterstreichen versucht, der Stallman wegen dessen dogmatischer Haltung mit dem Begriff des „Zeloten“[543] bezeichnet und das Gleichnis vom Kathedralenbau in der Softwareentwicklung[544] geprägt hat – auch dieser Raymond erscheint mit seinem auf Zen anspielenden Werk „The Art of UNIX-Programming“ nicht gerade ideologiefrei:

> “To do the Unix philosophy right, you have to be loyal to excellence. You have to believe that software design is a craft worth all the intelligence, creativity, and passion you can muster. [...] Software design and implementation should be a joyous art, a kind of high-level play. If this attitude seems preposterous or vaguely embarrassing to you, stop and think; ask yourself what you’ve forgotten. Why do you design software instead of doing something else to make money or pass the time? You must have thought software was worthy of your passion once... To do the Unix philosophy right, you need to have (or recover) that attitude. You need to care. You need to play. You need to be willing to explore.”[545]

Die klischeehafte Darstellung von Zen dient hier zur Begründung für die Forderung nach absoluter Hingabe, eine Forderung die, träte sie in beruflichen Zusammenhängen auf, so absurd erscheinen müßte wie der Squiggle der Wal-Mart-Mitarbeiter. Raymond zieht zahlreiche derartige Parallelen zwischen Hacken und Zen-Buddhismus; so beschreibt er in seiner Veröffentlichung „Rootless Root“[546] Unix-Praktiken in der Form von Koans und empfiehlt in seiner Anleitung „How to Become a Hacker“ ausdrücklich das Studieren von

---

[541] TORVALDS: Just for..., a.a.O., S. 205.

[542] Vgl. MOODY, a.a.O., S. 177.

[543] RAYMOND: Homesteading the..., a.a.O., S. 68ff.

[544] Vgl. RAYMOND: The Cathedral..., a.a.O.

[545] Vgl. RAYMOND: The Art..., a.a.O., Kapitel 1, Sektion 9, Online unter: http://www.catb.org/~esr/writings/taoup/html/ch01s09.html [Aufgerufen am 29.01.2005].

[546] Vgl. RAYMOND, Eric Steven: Rootless Root. The Unix Koans of Master Foo. Online veröffentlicht unter: http://www.catb.org/~esr//writings/unix-koans/ [Aufgerufen am 29.01.2005].

Zen und das Lernen eines Kampfsports: „The mental discipline seems similar in important ways“.[547]

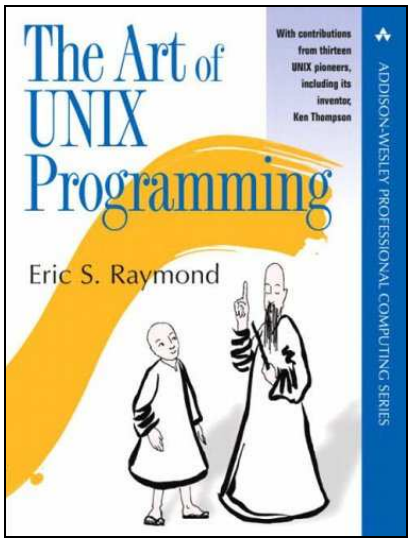


Abbildung 25 Titelbild von Eric S. Raymonds „The Art of Unix Programming“

Keinem der genannten Vertreter der GNU/Linux-Szene kann ernsthaft unterstellt werden, sie hätten religiös-missionarische oder ideologische Ambitionen. Allerdings könnte eine weitergehende Untersuchung hier möglicherweise vorbewußte Motive des Community-Building, der Entwicklung einer Organisationskultur nach ideologischem Muster aufdecken.[548] Es ist nicht zufällig, wenn Journalisten feststellen, daß „der Hackerkodex immer auch mit Ritterlichkeit und dem Weg des Samurai zu tun hat“[549] oder wenn IBM zu GNU/Linux Werbespots mit quasi religiösem Inhalt vorführt:

> „In dem Werbespot sieht man einen Jungen, über den eine Stimme aus dem Hintergrund sagt, er könne alles lernen, was man ihm beibringt. Danach tritt eine Heerschar von Prominenten auf, von der Box-Legende Muhammad Ali über die Regisseurin Penny Marshall hin zu einem Professor der Harvard-Universität. Sie alle teilen dem Jungen eine Lebensweisheit mit. Am Ende stellt sich heraus, daß der Junge „Linux“ heißt.“[550]

---

[547] RAYMOND: How to..., a.a.O., S. 207. Unter „Points for Style“ wird außerdem empfohlen: Eine Verbesserung der Rhetorik, das Lesen von Science Fiction, das Erlernen eines Musikinstruments und das Entwickeln eines Sinns für Kalauer.
Ein Punkt ist bei alledem anzumerken: Eric S. Raymond hat nur verhältnismäßig wenig Code zum GNU/Linux-System beigesteuert (Teile der Programme „fetchmail“, „ncurses“ und einige Emacs-Entwicklungen). Beiträge zum Linux-Kernel gibt es von ihm nicht. Seine „Kunst des Programmierens“ schätzen Kenner deswegen bei weitem nicht so hoch ein wie Stallmans Programme, die etwa bewertet werden mit: „brillant konstruiert, elegant geschrieben und macht süchtig“ (STOLL, a.a.O. S.35). Der einzige „Kampfsport“ aber, den Stallman je getrieben hat, ist Volkstanz, und den mußte er wegen eines Knieschadens nach wenigen Jahren aufgeben.

[548] So widerspricht die Beschwörung des disziplinarischen Verhältnisses zwischen Meister und Schüler im Zen-Buddhismus deutlich dem von HIMAMEN betonten partnerschaftlichen Umgang von Experte und Neuling in der „Netz-Akademie“. Vgl. HIMAMEN, a.a.O., S. 98.

[549] MAUS, a.a.O.

[550] LINDNER, Roland: Aktie im Blick: IBM. Linux lebt. Frankfurter Allgemeine Zeitung, 13.09.03, S. 21 (Wirtschaft). Beschrieben wird eine in der zweiten Septemberwoche 2003 gestartete Werbekampagne von IBM.

3.4.3.2 Star Wars

Es ist verblüffend, in welchem Ausmaß immer wieder die Star-Wars-Filme[551] von George Lucas der Hackerszene als Wertematrix dienen. So sollen Unix-Administratoren bereits Mitte der 80er[552] ihre Büros mit einem Plakat geschmückt haben, auf dem ein kleines Rebellen-Raumschiff vom Typ „X-Wing“ hinter sich einen großen explodierenden „Todesstern“ zurückläßt[553] – nur daß auf dem X-Wing-Flieger die Bezeichnung „4.2 BSD“ zu lesen war und der Todesstern durch ein brennendes AT&T-Logo symbolisiert wurde.[554]

Folgt man der Symbolik, so empfanden sich die Anhänger der universitär weiterentwickelten Unix-Variante BSD als Rebellen, als Kämpfer für die „Freiheit der Galaxis“ gegen den mächtigen Monopolisten AT&T, der mit seiner juristischen Inanspruchnahme des Unix-Codes das böse Imperium verkörperte. Das Logo von AT&T trägt noch heute den Spitznamen „Death Star“ – „Todesstern“.[555]

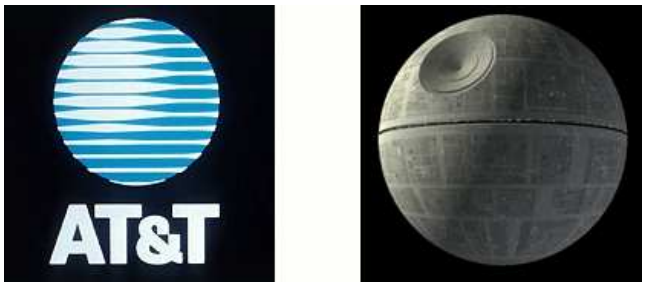


Abbildung 26: Das Logo der Firma AT&T und der“ Todesstern“ im Film „Star Wars“[556]

Diese dualistische Weltsicht dürfte bei den meisten Kennern der GNU/Linux-Gemeinschaft ein Déjà-vu verursachen: Heute übernimmt Microsoft die Rolle des Imperiums und Bill Gates den Part des „Darth Vader“, des Hackers, der sich von der dunklen Seite der Macht (vom Geld) hat verführen lassen. Zahlreiche Parodien im Internet stellen diesen Zusammenhang her.[557]

---

[551] Zu deutsch: Krieg der Sterne, bestehend aus: Star Wars (1977), The Empire Strikes Back (1980), The Return of the Jedi (1983), The Phantom Menace (1999), Attack of the Clones (2002) und Revenge of the Sith (2005, angekündigt). Die im vorliegenden Kapitel beschriebene Analogie bezieht sich vor allem auf den ersten Teil.

[552] Dies muß nach 1983 gewesen sein, denn in diesem Jahr erschien BSD-Unix 4.2 und zum gleichen Zeitpunkt begann AT&T seine Version von Unix selbst zu vermarkten und rechtlich zu schützen.

[553] Die Zerstörung des mondgroßen „Todessterns“ des (bösen) Imperiums durch einen solchen Jet der (guten) Rebellen markiert das Happy-End des ersten Star Wars-Films.

[554] Vgl. RAYMOND, Eric Steven: The New Hacker's Dictionary. Boston: MIT-Press, 1996. Zitiert nach der Online-Ausgabe unter: http://www.catb.org/~esr/jargon/html/D/Death-Star.html [Aufgerufen am 31.01.2005].

[555] Ebd.

[556] Quelle der Abbildungen: Das AT&T und Logo findet sich auf zahlreichen Webseiten, so auch auf: http://www.thetech.org/nmot/detail.cfm?id=19&st=awardDate&qt=1993&kiosk=Off
Das Bild des „Todessterns“ stammt nicht aus dem Film, sondern aus einem davon inspirierten Modellbausatz. Es fand sich bei: http://www.blenderwars.com/models/images/dstarlarge.jpg [beide aufgerufen am 29.01.2005].

[557] Die m.E. pointierteste Darstellung dieser Analogie findet sich bei: MUCHNICK, Avi: Microsoft nearing Completion of Death Star. In: BBSpot: 28.05.2002. Online veröffentlicht unter: http://bbspot.com/News/2002/05/deathstar.html [Aufgerufen am 26.01.2005]. Ein weiteres Beispiel findet sich unter:
The Net wars trilogy. http://www.eleves.ens.fr:8080/home/madore/computers/netwars.html [Aufgerufen am 26.01.2005].

Dabei ist Microsoft bereits die dritte Besetzung für die Rolle des übermächtigen Gegners der Hackerbewegung; die Premiere übernahm Anfang der 60er die Firma IBM. So stellt LEVY die Angestellten des damaligen Quasi-Monopolisten als Schlipsträger in Schwarzweiß dar, im Gegensatz zur Schmuddeligkeit der Hacker (Rebellen), und auch seine Darstellung der Firma erinnert an Star Wars: „As IBM's dominance of the computer field was established, the company became an empire unto itself, secretive and smug."[558]

ZIMMERMANN sieht in diesem Feindbild eine wichtige Funktion: „Hier entwickelt sich das Bild einer monopolisierten IT-Industrie als Gegner von FS/OS, dessen Bedeutung von Sprechern der Bewegung zwar gern heruntergespielt wird, welches jedoch gerade im Prozeß der Identitätsstiftung einer sozialen Bewegung von hoher Bedeutung ist."[559] Die Identität, die die GNU/Linux-Gemeinschaft in der öffentlichen Wahrnehmung aus ihrer Konfrontation mit Microsoft bezieht, wurde bereits in der Einleitung dieser Arbeit deutlich gemacht.[560]

Die beiden programmatischen Vordenker von GNU/Linux, Stallman und Raymond, beziehen sich immer wieder auf das Star-Wars-Universum. So heißt es etwa in Raymonds Essay „Revenge of the Hackers"[561] als Ausblick:

> "The path will be long, and it won't be easy. But I think the hacker community, in alliance with it's new friends in the corporate world, will prove up to the task. And, as Obi-Wan Kenobi might say, 'the Source will be with us'."[562]

Wie bei Schlußworten dieser Art üblich, wird hier an den Zusammenhalt, die Gruppenidentität appelliert, wobei sich Raymond auf eine Star-Wars-Figur, den Jedi-Ritter „Obi-Wan Kenobi" bezieht. Allerdings geht er dann darüber hinaus, indem er den typischen Segenswunsch des Filmhelden verdreht: Die übernatürliche Kräfte verleihende „Macht" („The Force") macht er in einem Kalauer zu „Quellcode" („The Source"). Im Zusammenspiel mit Star Wars gewinnt das Schreiben von Software höhere Bedeutung.

Stallman bezieht sich ebenfalls oft auf die Filme, was für Kenner des Films oft amüsante Formen annimmt. So erklärte Stallman, als ihm auf der kalifornischen Linux-World Show 1999 der „Linus Torvalds"-Preis verliehen wurde, seinem erstaunten Publikum: „'Giving the Linus Torvalds Award to the Free Software Foundation is a bit like giving the Han Solo Award to the Rebel Alliance."[563] (Für Nicht-Kenner: Die von Harrison Ford gespielte Figur des Han Solo trat den Rebellen als Letzter bei.)

Die Filmfiguren werden gelegentlich auch benutzt, um die Hierarchie der GNU/Linux-Anführer abzubilden. So berichtet WILLIAMS von einem T-Shirt, das bei GNU/Linux-

---

[558] LEVY, a.a.O., S. 42. Die Wortwahl ist möglicherweise nicht zufällig: Das Buch wurde zur Zeit des größten Erfolgs von Star Wars verfaßt. Vgl. auch die Darstellung der „Diktatur" von IBM in einem Werbefilm für den Apple MacIntosh aus dem Jahre 1984 unter: WIKIPEDIA (engl.): 1984 (television commercial). http://en.wikipedia.org/wiki/1984_(television_commercial) [Aufgerufen am 31.01.2005].

[559] ZIMMERMANN, Thomas. Open Source und Freie Software – Soziale Bewegung im virtuellen Raum. In: GEHRING; LUTTERBECK (Hsg.), a.a.O., S. 353-368, hier S. 360.

[560] So geht aus einer oberflächlichen Untersuchung von 386 Zeitungsartikeln in FAZ und Süddeutscher Zeitung von 1994 bis 2003 hervor, daß von den 128 Artikeln, die immerhin GNU/Linux und Freie bzw. Open-Source-Software zum Hauptthema ihrer Berichterstattung machen, gerade mal 31 ohne die Erwähnung von Microsoft oder Windows auskommen.

[561] Der Titel selbst ist eine Anspielung auf „Return of the Jedi". Vgl. RAYMOND: Revenge of..., a.a.O.

[562] Ebd., S. 191.

[563] WILLIAMS, a.a.O., S. 60.

Vorführungen verkauft wurde und prominente Mitglieder der Bewegung als Star-Wars-Figuren zeigte. Stallman habe deutliches Unbehagen darüber geäußert, daß man ihn dem kleinen Roboter R2D2 nachempfunden hatte, seinen Rivalen Torvalds aber als strahlenden Helden Skywalker abbildete.[564]

Doch auch, wenn Stallman über Strategie und Zukunft seiner Freien-Software-Bewegung spricht, liegt Star Wars nicht fern:

> "Yoda's philosophy ("There is no 'try'") sounds neat, but it doesn't work for me. I have done most of my work while anxious about whether I could do the job, and unsure that it would be enough to achieve the goal if I did. But I tried anyway, because there was no one but me between the enemy and my city. Surprising myself, I have sometimes succeeded. Sometimes I failed; some of my cities have fallen. Then I found another threatened city, and got ready for another battle. Over time, I've learned to look for threats and put myself between them and my city, calling on other hackers to come and join me."[565]

Im zitierten Textausschnitt, ebenfalls ein Schlußwort, bezieht sich Stallman auf die Figur Yoda, den „Jedi-Lehrer" in den Filmen, und reflektiert zunächst über dessen Motto: „Nicht versuchen! Tu es oder tu es nicht. Es gibt kein Versuchen". Bemerkenswert ist Stallmans anschließende Darstellung seines eigenen Kampfes: Ganz offensichtlich sieht er sich als Ritter, der „seine Städte" vor der Einnahme des Feindes bewahrt. Auch hier werden mythologische Motive eingesetzt, um die Werte der Gemeinschaft zu beschwören; der gemeinsame Kanon aber heißt: „Star Wars".

3.4.3.3 Hacking

Die Hackertechnik der Rekursivität, also der Erklärung einer Tatsache aus sich selbst heraus, macht auch vor dieser Arbeit nicht halt: Denn auch die hier dargestellten Entwicklungen und Personen stellen selbst wiederum die Helden und Legenden dar, auf die sich die Szene immer wieder bezieht.

So beschreibt zum Beispiel Torvalds in seinem autobiografischen Buch „Just for fun" Stallmans Kampf mit Symbolics und leitet dies mit den Worten ein: „Stallman, der es verdienen würde, daß man ihm zu Ehren ein Denkmal für die Schaffung der GPL errichtete".[566] Raymond wiederum hat Torvalds Arbeits- und Führungsstil in seinem Werk „The Cathedral and the Bazaar" zum Thema gemacht und spricht von einem dieser Prinzipien sogar als „Linus' Law".[567]

Sowohl Raymond wie auch Stallman verweisen immer wieder auf die Hacker-Ursprünge. So beschreibt etwa Stallman seine Erfahrung einer paradiesischen[568] Hacker-Gemeinschaft am MIT: „I had the good fortune in the 1970's to be part of a community of programmers who shared software. Now, this community could trace its ancestry essen-

---

564 Vgl. WILLIAMS, a.a.O., S. 175.

565 STALLMAN: The GNU-Project, a.a.O., S.30.

566 TORVALDS: Just for..., a.a.O., S. 207.

567 Vgl. RAYMOND: The Cathedral..., a.a.O., S. 30.

568 Vgl. GREVE, Georg C. F.: Geschichte und Philosophie des GNU Projektes. Vortrag an der Universität Paderborn, 05. 12.1998. Online veröffentlicht unter: http://www.gnu.org/philosophy/greve-clown.de.html [Aufgerufen am 09.12.2204].

tially back to the beginning of computing."[569] Und Raymond beschwört diese Zeit, als sei sie ein Schöpfungsmythos: „In the beginning, there were Real Programmers".[570]

Mit seinen Texten zur Hacker-Geschichte wie „A Brief History of Hackerdom", „Revenge of the Hackers", „How to Become a Hacker" und der Herausgabe der Wörter- und Anekdotensammlung „The New Hackers Dictionary" sorgt Raymond außerdem für die Erhaltung von Traditionen, was für die Herstellung einer Gruppen-Identität wichtig ist: „Aus den Anfängen der Hackerkultur entwickelten sich diverse spezifische Sprachnormen, Normen und Regeln, die den Teilnehmern eine pragmatische und attraktive Identität mit hohem Mobilisierungspotential bietet."[571]

Und das mit Erfolg: Die Einschätzung, Teil einer „Gemeinschaft von Hackern" zu sein, ist in freien Softwareprojekten weit verbreitet und wird von den Mitgliedern als wichtige Quelle für ihr Selbstbewußtsein empfunden: So stellen LAKHANI und WOLF fest: „Respondents to our survey indicated a strong sense of group identification with 42% indicating that they ‚strongly agree' and another 41% ‚somewhat agree' that the hacker community is a primary source of their identity."[572]

Insgesamt ist der Einsatz des Koordinationsinstrumentes „Organisationskultur" in der betriebswirtschaftlichen Organisationslehre umstritten. So weisen KIESER und KUBICEK darauf hin, daß die Flexibilität der Mitarbeiter dadurch herabgesetzt werden kann. Das heißt: Ein überzeugter „Rebell gegen den Todesstern" hat möglicherweise Schwierigkeiten, an eher konservativen Projekten zu arbeiten.[573] Außerdem kann die Anpassungsfähigkeit der Organisation selbst unter der Organisationskultur leiden – etwa, wenn Ereignisse eintreten, die eine radikale Neuorientierung nötig machen.[574]

Im folgenden und letzten Kapitel dieser Arbeit soll eine solche Neuorientierung kurz beschrieben werden: Die Erkenntnis des enormen wirtschaftlichen Potentials von freier Software und die Einführung des Begriffs „Open Source".

3.5 Der Begriff "Open Source"

Die Gemeinschaft der Entwickler am Linux-Kernel entstand unabhängig von Stallmans GNU-Projekt. Es gab nur zwei Gemeinsamkeiten: Zum einen die *Lizenz*, die GNU GPL, unter die Torvalds den Kernel 1992 gestellt hatte und zum anderen die *Verwendung* von GNU-Programmen. Der Kernel selbst als unterste Schicht des Betriebssystems verteilt nur die Zugriffe auf Rechenzeit, Arbeitsspeicher und Hardware, z.B. die Festplatte. Erst durch weitere Systemsoftware (wie z.B. die „bash", eine Text-Benutzerschnittstelle) wird es möglich, Befehle einzugeben und Programme zu verwenden. Bei dem Aufbau eines funktionsfähigen Systems auf diesem Kernel war man also auf GNU-Software angewiesen, aber auch auf viele weitere Programme, die wie Linux unabhängig von GNU entstanden

---

569 STALLMAN: Free Software: Freedom..., a.a.O., S. 157.
570 RAYMOND: A Brief..., a.a.O., S. 3.
571 ZIMMERMANN, a.a.O., S. 360.
572 LAKHANI; WOLF, a.a.O., S. 12.
573 Vgl. KIESER; KUBICEK, a.a.O., S. 124f.
574 Vgl. KIESER; KUBICEK, a.a.O., S. 125.

waren:[575] Freigegebene Werkzeuge aus der Unix-Version BSD, die grafische Schnittstelle „X“, und wichtige Anwendungen wie BIND[576], Apache[577] und Sendmail[578].

Die meisten Linux-Programmierer empfanden sich also lediglich als Anwender von GNU-Software und nicht als Mitarbeiter des GNU-Projekts. Als letztere waren sie auch nicht gewünscht: Ende 1991 hatte Stallman entschieden, daß Linux keine Basis für sein eigenes System werden sollte; er schätzte es als technisch ungenügend ein.[579] Es ist im Nachhinein kaum zu glauben, daß Stallman, der seine eigenes Projekt als soziale Bewegung sieht und dem als regelmäßigen Besucher von Science-Fiction-Conventions auch die Bedeutung von „Fandom“ bekannt gewesen sein muß, die soziale Dynamik von Linux nicht erkannte.[580] Statt dessen setzte er weiter auf die Entwicklung der eigenen Kernel-Alternative „Hurd“ und verzettelte sich: „They moved away from the goal of doing a production-ready operating system to doing operating-system research“.[581] Den zunehmenden Erfolg und die Verbreitung des Konkurrenzsystems Linux verfolgte Stallman währenddessen mit Unbehagen, obwohl es in der Sache um das Gleiche ging – ein freies Unix: „Like Winston Churchill watching Soviet troops sweep into Berlin, Stallman felt an understandable set of mixed emotions when it came time to celebrate the Linux ‚victory’.“[582]

3.5.1 Die Kommerzialisierung von GNU/Linux

Ende 1992 entstanden die ersten GNU/Linux-Distributionen, Programmpakete in Pappkartons, wie sie heute noch in Buchhandlungen und Elektronikmärkten verkauft werden. Diese Zusammenstellungen sorgten dafür, daß man nicht mehr mühsam alle Teile des Systems aufeinander abstimmen mußte – sie waren ein Zeichen dafür, daß GNU/Linux begann, für Nicht-Hacker interessant zu werden, und auch für den kommerziellen Einsatz.

Abbildung 27: Logos von Linux-Distributoren

---

[575] Einen Einfluß der GNU-Idee auf die jeweiligen Entwickler gab es freilich, allerdings gehörten sie weder zum GNU-Projekt, noch stellten sie ihre Software unter die GNU GPL.

[576] Domain Name Server, zur Namensauflösung im Internet

[577] Webserver, zur Auslieferung von Webseiten an Browser.

[578] Mailserver zum Versand und der Weiterleitung von E-Mails.

[579] Vgl. WILLIAMS, a.a.O., S. 145.

[580] Microsoft-Gründer Bill Gates wird gelegentlich vorgeworfen, die Bedeutung des Internets zu spät erkannt zu haben. Dies kann man im vorliegenden Zusammenhang auch Stallman vorhalten. Die einzelnen Bastler aus privaten Motiven wurden durch ihren Zusammenschluß in Newsgroups zu einer Massenbewegung, die das System allein durch die Zahl ihrer Köpfe schneller technisch vorwärts trug, als dies einer zentralen, „bewußten“ Instanz mit einem prinzipiell überlegenen Ansatz in der gleichen Zeit möglich war.

[581] WILLIAMS, a.a.O., S.146. WILLIAMS zitiert damit Eric S. Raymond, der damals Mitarbeiter des GNU-Projektes war. Stallman tappte damit wie viele Programmierer in die „Perfektionismus-Falle"; ausgerechnet sein früherer Erzfeind „Symbolics“ scheiterte an dem gleichen Problem. Vgl. GRAYLIN et. al., a.a.O., S. 18.

[582] WILLIAMS, a.a.O., S. 147.

Die ersten Firmen, die diesen Markt bedienten, trugen Namen wie „SLS“, „Slackware“, „Yggdrasil“ und in Deutschland „S.u.S.E.“[583]. Kurze Zeit darauf wurden die beiden heute wichtigsten Distributionen gegründet: „Red Hat“ und „Debian“. Red Hat wurde herausgegeben von einem Privatunternehmen, dem heutigen Marktführer.[584] Debian dagegen war der Versuch, das Entwicklungsmodell des Linux-Kernels auf ein ganzes System zu übertragen. Debians Gründer, Ian Murdock, war damit vielleicht der erste, der versuchte, die sozialen Grundlagen des Erfolgs von Linux für ein eigenes Projekt nutzbar zu machen:

> “What we really ought to do is we ought try to take the model that Linux has pioneered, intentionally or not, and try to get the same benefits from that for building the system around it.” [585]

Die Ankündigung seines Projekts in der Newsgroup comp.os.linux stieß auf hohe Resonanz. Um das Modell der Linux-„Lieutenants“ übernehmen zu können, entwickelte Murdock das sogenannte „Paketsystem“[586] von Debian: Alle Teile des Betriebssystems und sogar einzelne Programme wurden in Einzelpakete unterteilt, um eine möglichst flache Hierarchie von Zuständigkeiten zu erzeugen:

> “Debian would be based on the idea of a package, and all these people who wanted to work on it could then take responsibility for all these different packages. [...] Other people taking on subtaskings is very much how Linux worked.” [587]

Einer derjenigen, die auf Murdocks Ankündigung reagierten, war Richard Stallman. Er hatte 1993 eingesehen, daß sein GNU-Projekt mit der Entwicklungsgeschwindigkeit von Linux nicht mithalten konnte. Seine Mitarbeiter waren mehr und mehr überfordert von den Ansprüchen, die die technisch immer versierteren Linux-Programmierer an die GNU-Entwicklerwerkzeuge zu stellen begannen – verärgerte Linux-Anhänger drohten bereits damit, die Weiterentwicklung wichtiger GNU-Programme selbst zu übernehmen.[588] Stallman mußte befürchten, daß sein Projekt früher oder später in Linux aufgehen und er selbst marginalisiert würde. Debian sah er deswegen als Möglichkeit, einen Fuß in die Tür zu bekommen, wie er in einer Mail an Murdock mitteilte:

> “He said the Free Software Foundation was starting to look closely at Linux and that the FSF was interested in possibly doing a Linux system, too. Basically, it looked to Stallman like our goals were in line with their philosophy.”[589]

Murdock willigte ein. Damit er sein Projekt voranbringen konnte, erhielt er nun ein Honorar von Stallmans Free Software Foundation, diese machte das Projekt bekannt und

---

583 Akronym für „Software und System-Entwicklung“.

584 Red Hat hat ca. 20 bis 30 Millionen Nutzer, sein Gewinn lag im dritten Quartal 2004 bei 10,9 Millionen US-Dollar. Vgl. Frankfurter Allgemeine Zeitung: Linux findet Freunde in Unternehmen. Auch kleinere Anbieter wie Red Hat kommen besser ins Geschäft. 26.06.03, S. 15 (Wirtschaft). Und: LINUX Magazin: Red Hat steigert Gewinn. Ausgabe 03/05. München: Linux New Media AG, S. 21.

585 MOODY, a.a.O., S. 89. Zitiert wird hier Ian Murdock, den Gründer des Debian-Projekts. Benannt wurde die Distribution nach dessen Vornamen und dem seiner Frau, Deborah.

586 Wegen seiner Leistungsfähigkeit bei Updates und Fehlerkorrektur findet das Paketsystem noch heute häufig lobende Erwähnung. Es ist ein weiteres Beispiel dafür, wie technische Aspekte des Systems durch organisatorische Überlegungen geprägt werden.

587 MOODY, a.a.O., S. 90. Zitiert wird hier Ian Murdock.

588 Vgl. WILLIAMS, a.a.O., S. 148f.

589 WILLIAMS, a.a.O., S. 145. Zitiert wird hier Ian Murdock.

übernahm auch den Vertrieb der Software.[590] Gleichzeitig flossen die strikten Vorstellungen Stallmans über freie Software in die Distribution ein, in völliger Übereinstimmung mit Ian Murdock und den übrigen Entwicklern, wie im wenig später erschienenen „Debian Manifesto“ deutlich wird:

> “By the simple fact that they [Free Software Foundation] will be distributing it, a message is sent to the world that Linux is not a commercial product and that it never should be, but that this does not mean that Linux will never be able to compete commercially. [...] The time has come to concentrate on the future of Linux rather than on the destructive goal of enriching oneself at the expense of the entire Linux community and its future.”[591]

Die Zusammenarbeit zwischen der Free Software Foundation und der Debian-Distribution ging allerdings nur drei Jahre lang gut: Stallman Führungsstil mißachtete die Freiwilligkeit seiner Mitarbeiter; der spätere Debian-Leiter Bruce Perens spricht von „Micromanagement“,[592] d.h. Stallman neigte dazu, in kleinste Fragen hineinzuregieren.

> “Richard wanted to have a part in the technical direction, and the developers on the project didn’t really want a boss [...] they were all volunteers. If they wanted to take any direction it would be from someone who was working with them on the project. [...] We took issue with some of his technical direction because we felt it would lower the performance of the system and make it take up more disk space. We decided that we just could not live with the FSF having a right to tell us what to do, and so we split off our own organization.”[593]

Für erheblichen Unmut zwischen Stallman und vielen Linux-Entwicklern sorgte zudem seine Forderung, das System fortan „LiGNUx“ zu nennen – eine Referenz an die Bedeutung der GNU-Software im Betriebssystem. Zwar willigte das Debian-Projekt als Kompromiß ein, die Distribution „Debian GNU/Linux“ zu nennen, letztlich führte Stallmans Verhalten aber 1996 zum Bruch zwischen beiden Projekten.

Debian wird heute getragen von etwa 1.000 Freizeitprogrammierern weltweit, ist vor allem auf Servern sehr weit verbreitet und dient als technische Basis zahlreicher weiterer Distributionen wie „Xandros“, „Knoppix“ und „Ubuntu“. Debians Community-Modell wurde zudem mehrfach imitiert: So gründete Red Hat 2003 das „Fedora-Projekt“, um den Input von Freiwilligen zu nutzen; die französische Distribution Mandrake geht mit „Mandrake-Community“ den gleichen Weg, ebenso wie Novell seit 2005 mit „OpenSuse“.

Trotz des formalen Bruchs mit Stallman blieb Debian seiner Idee von freier Software enger verbunden als jede andere Distribution: Debian besteht vollständig aus freier Software. Das Projekt setzt damit ein Zeichen gegen die inzwischen verbreitete Geschäftspraktik, freie Programme mit kommerziellen zu mischen, um ähnlich hohe Lizenzgebühren verlangen zu können, wie Konkurrent Microsoft.

---

590 Vgl. WILLIAMS, a.a.O.., S. 147, sowie MOODY, a.a.O., S. 90. MOODY zitiert Murdock mit den Worten: „[Stallman's] early interest was critical in [...] giving it an air of legitimacy [...] I really think that, more than anything else in those early days, is what caused people to take notice and get involved.“

591 MURDOCK, Ian: A Brief History of Debian. Appendix A: The Debian Manifesto. 01.06.1994. Online veröffentlicht unter: http://www.debian.org/doc/manuals/project-history/ap-manifesto.en.html [Aufgerufen am 02.02.2005].

592 WILLIAMS, a.a.O., S. 150.

593 MOODY, a.a.O., S. 92. MOODY zitiert den zweiten Debian-Leiter, Bruce Perens.

3.5.2 Die Open-Source-Bewegung

Obwohl die Free-Software-Bewegung mit ihrem Versuch gescheitert war, über die Distribution Debian signifikanten Einfluß auf die Linux-Entwickler zu gewinnen, bildeten die Begriffe und Texte Richard Stallmans lange Zeit die einzige theoretische Grundlage für die Gemeinschaft. Torvalds selbst hatte keinerlei Interesse an programmatischen Entwürfen. In diese Lücke trat 1997 der frühere Mitarbeiter des GNU-Projekts, Eric S. Raymond. Mit seinem Vortrag "The Cathedral and the Bazaar. Why does the Linux development model work?"[594] beim 4. Linux-Kongreß am 22.05.1997 in Würzburg stellte er erstmals seine Thesen zu Torvalds' Vorgehensweise und Managementstil der Öffentlichkeit vor.[595]

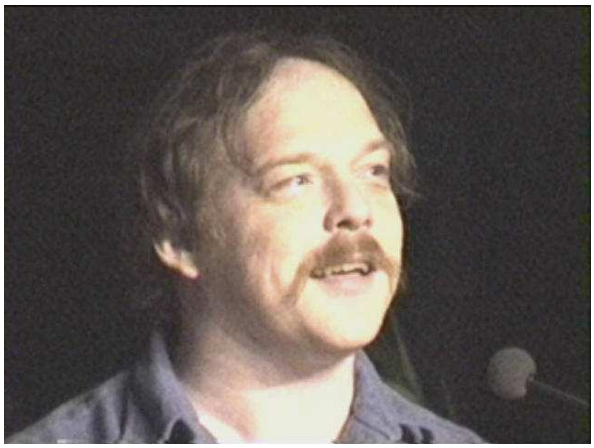


Abbildung 28 Eric S. Raymond im Jahr 1999[596]

In den Mittelpunkt stellte er die Überlegung, daß bisher Software nach dem Modell einer *Kathedrale* programmiert worden sei: Von erfahrenen Baumeistern, nach genauen Plänen und über einen langen Zeitraum bis zur Vollendung. Als Beispiel für diese Art Softwareentwicklung nannte er kommerzielle Software sowie das GNU-Projekt.[597] Die gemeinschaftlichen Entwicklungsmethode Torvalds' sei dagegen ein neuer Entwicklungsstil, ein *Basar*-Stil, der seine alten Überzeugungen ins Wanken gebracht habe:

> "I believed that the most important software [...] needed to be built like cathedrals, carefully crafted by individual wizards or small bands of mages working in splendid isolation, with no beta to be released before its time. Linus Torvalds's style of development – release early and often, delegate everything you can, be open to the point of promiscuity – came as a surprise. No quiet, reverent cathedral-building here – rather, the Linux community seemed to resemble a great babbling bazaar of differing agendas and approaches."

In seiner weiteren Argumentation beschrieb Raymond die Vorgehensweise Torvalds' bei der Linux-Entwicklung und überprüfte deren Wiederholbarkeit durch ein Experiment: Er startete ein eigenes, kleineres Projekt, bei der ein Programm zur automatischen Abfrage

---

[594] Eine Kurzzusammenfassung der damaligen Rede findet sich online unter: http://www.linux-kongress.org/1997/raymond.html [Aufgerufen am 02.02.2005].

[595] Vgl. WILLIAMS, a.a.O., S. 159. Ein vorheriger Vortrag vor Freunden und Nachbarn wird hier – anders als bei MOODY – nicht als Erstveröffentlichung verstanden.

[596] Bei der Linux World Conference & Expo (LWCE) in San Jose, 1999. Foto von Doran L. Barton. Veröffentlicht online unter: http://www.fozzilinymoo.org/events/1999-LWCE/images/Panel-Eric.jpg [Aufgerufen am 02.02.2005].

[597] Vgl. RAYMOND: The Cathedral..., a.a.O., S.28f.

von E-Mail gemeinschaftlich erstellt werden sollte.[598] Auf dieser Basis stellte er eine Reihe von Regeln für den „Basar-Stil“ auf; er empfahl unter anderem:

- keine Scheu zu haben, Code aus anderen Projekten wiederzuverwenden,
- keine Scheu zu haben, vorhandenen Code wegzuwerfen,
- seine Nutzer als Helfer und Mitentwickler zu betrachten und
- häufige Vorversionen der Software zu veröffentlichen.

Als Raymonds berühmteste Regel gilt dabei der als „Linus' Law“ bezeichnete Satz: „Given enough eyeballs, all bugs are shallow“[599], was übersetzt werden könnte mit: „Vor den Augen vieler verflüchtigen sich alle Fehler“. Gemeint ist die gegenseitige Fehlerkorrektur, die um so effektiver wird, desto mehr Menschen sich daran beteiligen.[600]

Raymonds Rezept zur Wiederholung des Linux-Erfolgs stieß im damaligen Dot-Com-Boom auf erhebliche Resonanz. Das deutsche „Linux-Magazin“ machte den Vortrag zum Aufmacher und titelte „Jungbrunnen für neue Software. Wie die Linux-Methode funktioniert“[601] als sei das Ganze eine neuartige Diät. Das bedeutendste Echo auf die von Raymond vielfach wiederholte Rede aber kam aus Kalifornien: Im Januar 1998 beschloß die Firma Netscape, nach dreijährigem erfolglosen Konkurrenzkampf mit Microsoft um den Markt für Webbrowser, ihr eigenes Produkt „Netscape Communicator“ als freie Software freizugeben. Geschäftsführer Barksdale bezeichnete „The Cathedral and the Bazaar“ als den wichtigsten Grund für diese Entscheidung.[602] Bis dahin als Geschäftsgeheimnis gehütete 30 Millionen Codezeilen wurden in das öffentliche Eigentum des „Mozilla“ getauften Projekts überführt – man erhoffte sich so die Rückgewinnung von Marktanteilen.[603]

Die Entscheidung von Netscape wiederum weckte bei Raymond den Wunsch, für freie Software wie Linux und das von ihm beschriebene Entwicklungsmodell eine eigene, neue Bezeichnung zu finden. Ziel war es, das durch die Netscape-Entscheidung hervorgerufene Medieninteresse auszunutzen, um bei weiteren Unternehmen für eine Übernahme des Konzepts zu werben. Der Begriff “freie Software” stand dem entgegen, meinte damals auch Debian-Aktivist Bruce Perens: “The word ‘free’ is somewhat intimidating to business in that they think they can’t make any money if it’s free.”[604]

Weniger als einen Monat nach der Entscheidung von Netscape, am 03.02.1998 veranstaltete Raymond deswegen ein Treffen mit GNU/Linux-Unternehmern und Anhän-

---

598 Das so entstandene Programm „fetchmail“ ist heute noch vielfach im Einsatz.

599 RAYMOND: The Cathedral..., a.a.O., S.30.

600 Der Vorteil der gemeinschaftlichen Entwicklungsmethode bei der Fehlersuche ist mittlerweile auch theoretisch untersucht und bestätigt worden. Vgl. CHALLET, Damien; LE DU, Yann: Closed source versus open source in a model of software bug dynamics. Online veröffentlicht unter: http://arxiv.org/abs/cond-mat/0306511 [Aufgerufen am 02.02.2005].

601 Vgl. Online-Ausgabe unter: http://www.linux-magazin.de/Artikel/ausgabe/1997/08 [Aufgerufen am 02.02.2005].

602 Vgl. WILLIAMS, a.a.O., S. 161.

603 Eine nennenswerte Rückgewinnung von Marktanteilen begann erst Ende 2004 mit dem Mozilla-Ableger Firefox, fast sieben Jahre nach Netscapes Entscheidung. Allgemein wird für die lange Wartezeit auch die Qualität des ursprünglich freigegebenen Codes verantwortlich gemacht; große Teile waren aus lizenzrechtlichen Gründen verstümmelt. Vgl. RAYMOND: The Cathedral..., a.a.O., S. 61ff.

604 MOODY, a.a.O., S. 166. Moody zitiert aus einem eigenen Interview mit Bruce Perens.

gern,[605] um einen anderen Begriff zu finden. Die Teilnehmer nannten „freeware“, „sourceware“ und „freed software“ – schließlich schlug Christine Peterson, Vizepräsidentin des Nanotechnik-Lobbyisten „Foresight“ den Begriff „Open Source“ vor, der auf allgemeine Zustimmung stieß. Bei einem Treffen der wichtigsten GNU/Linux-Entwickler,[606] dem „Freeware-Summit“ am 07.04.1998 wurde der Begriff dann mit neun gegen sechs Stimmen angenommen und ab sofort in allen Pressekontakten verwendet.[607] Der Veranstalter, der Verleger Tim O’Reilly, hatte Richard Stallman zu dem Treffen absichtlich nicht eingeladen – er befürchtete, daß dieser die Einigung auf den neuen Begriff verhindern würde.[608]

Abbildung 29 Bruce Perens, früherer Debian-Projektleiter, Autor der „Open Source Definition“.[609]

Als programmatischen Unterbau seiner neuen „Open-Source-Initiative“[610] wählte Raymond die „Debian Free Software Guidelines“, ein von dem damaligen Debian-Leiter Bruce Perens verfaßtes Manifest, das die Anforderungen an die Lizenzen der in der Distribution verwendeten freien Software festschrieb: „We wanted Debian to be 100 percent free software, and at the time we sort of knew what free software was from the philosophy of Richard Stallman and the Free Software Foundation.“[611] Die „Debian Free Software Guidelines“ wurden dann auf Bitten Raymonds umgeschrieben in die „Open Source Definition“, wobei Perens lediglich den Begriff „Debian“ aus dem Dokument entfernte.

---

605 Teilnehmer waren Larry Augustin (VA Research), John „maddog“ Hall (Digital), Sam Ockman (Silicon Valley Linux User Group) und Christine Peterson (Foresight Institute).

606 Unter anderen: Linus Torvalds (Linux), Michael Tiemann (Cygnus) John Gilmore (Cygnus), Guido van Rossum (Python), Paul Vixcie (Bind), Larry Wall (Perl) Eric Allman (Sendmail), Eric S. Raymond.

607 Vgl. WILLIAMS, a.a.O., S.164.

608 MOODY zitiert in diesem Zusammenhang Raymond: „Tim O'Reilly and the other co-organizers thought he would disrupt the effort to achieve a consensus from which we could go forward.“ MOODY, a.a.O., S. 166.

609 Quelle des Fotos: Homepgae von Bruce Perens: http://perens.com/ [Aufgerufen am 10.10.2005].

610 So der Name der Organisation, deren Präsident Raymond bis vor kurzem war. Vgl. Open Source Initiative will Aufgabenbereich ausdehnen. heise-online, 02.02.2005, 09:48 Uhr. http://www.heise.de/newsticker/meldung/55823 [Aufgerufen am 03.02.2005].

611 MOODY, a.a.O., S. 167.

So kommt es, daß die Begriffe „Open Source“ und „freie Software“ per Definition das Gleiche bezeichnen – nämlich ein Programm, das die von Stallman in seiner „Free Software Definition“ aufgeführten *vier Freiheiten* gewährleistet (siehe Kapitel 2.3.3.1).[612]

Noch im gleichen Jahr distanzierte sich Richard Stallman von dem Begriff: Er warf den Verfechtern vor, die Idee der Freiheit zugunsten kurzfristiger Ziele zu opfern. Auch wenn sich die offizielle Definition von „Open Source“ nicht von freier Software unterscheide, lade der Begriff doch zu Mißverständnissen ein – betont werde nicht mehr der Wert der Freiheit, sondern nur noch die Lesbarkeit des Quellcodes: „[...] the obvious meaning for the expression ‚open source software’ is ‚You can look at the source code.’“[613] Die Nutzer könnten aus dem Blick verlieren, daß es bei freier Software vor allem um ihr Selbstbestimmungsrecht gehe; sie könnten versucht sein, Software zu akzeptieren, die zwar den Quellcode offenlegt, Änderung und Weitergabe aber beschränkt:

> “Sooner or later these users will be invited to switch back to proprietary software for some practical advantage. Countless companies seek to offer such temptation, and why would users decline? Only if they have learned to *value the freedom* free software gives them, for its own sake. It is up to us to spread this idea – and in order to do that, we have to talk about freedom.”[614]

In seinen Texten und seinen Reden stellt Stallman deshalb „Open Source“ und „Free Software“ als zwei unterschiedliche Bewegungen dar:

> “People have been speaking of me in the context the Open Source movement. That’s misleading because I am not a member of it. I belong to the Free Software movement. In this movement we talk about freedom, about principle, about the rights that computer users are entitled to. The Open Source movement avoids talking about those issues, and that is why I am not joining it.”[615]

Ob unter „Open Source“ tatsächlich ein anderes Phänomen zu verstehen ist, als unter „freier Software“ scheint meines Erachtens aber fraglich: So hält die Open Source Initiative in ihrer Selbstbeschreibung ausdrücklich fest, daß der Begriff „Open Source“ lediglich Marketing-Zwecken dient – gemeint sei damit „freie Software“:

> “The Open Source Initiative is a marketing program for free software. It’s a pitch for “free software” on solid pragmatic grounds rather than ideological tub-thumping. The winning substance has not changed, the losing attitude and symbolism have.”[616]

---

612 Stallman sieht trotz gleicher Definition Unterschiede in der Interpretation. So habe die Open Source Initiative einige Lizenzen akzeptiert, die die FSF für proprietär hält: „The official definition of ‘open source software', as published by the Open Source Initiative, is very close to our definition of free software; however, it is a little looser in some respects, and they have accepted a few licenses that we consider unacceptably restrictive of the users. [...] it includes free software, but also includes semi-free programs such as Xv, and even some proprietary programs, including Qt under its original license (before the QPL).“ STALLMAN, Richard Matthew: Why „Free Software“ is better than „Open Source“. In: DIBONA, Chris; OCKMAN, Sam; STONE, Mark (Hrsg.): OpenSources. Voices from the Open Source Revolution. Sebastopol: O'Reilly, 1999, S. 55-60, hier: S.56. Die Zahl der zwischen FSF und OSI umstrittenen Lizenzen ist aber äußerst gering.

613 STALLMAN, Richard Matthew: Why „Free..., a.a.O., S.57.

614 Ebd., S.56.

615 STALLMAN, Richard M.: RMS responds. E-Mail an das Webforum des Online-Magazins Slashdot vom 28.6.1999, 08:03. http://slashdot.org/article.pl?sid=99/06/28/1311232&mode=thread [Aufgerufen am 2.2.05].

616 OPEN SOURCE INITIATIVE: Frequently Asked Questions. How is „open source“ related to „free software"? Online veröffentlicht auf: http://www.opensource.org/advocacy/faq.php [Aufgerufen am 02.02.2005].

Den Eindruck, daß es sich nur um unterschiedliche Worte für die gleiche Sache handelt, unterstützen auch die zahlreichen Wortkonstruktionen, mit denen wissenschaftliche Autoren politisch korrekt versuchen, beide Begriffe „gleichwertig“ zu verwenden. So finden sich Akronyme wie “***FS/OS***” (Free Software/Open Source), “***FOSS***” (Free and Open Source Software), “***FLOSS***” (Free/Libre Open Source Software) oder gar die Kombination “***Free Open-Source Software***”.[617] Bezeichnet wird so nicht nur die Software, sondern das gesamte Phänomen und alle beteiligten Akteure.[618] Der hohe Bedarf an einem einheitlichen Begriff, der sich in diesem Kauderwelsch ausdrückt, weckt dabei den Verdacht, dass die Dinge, die damit vereinheitlicht werden sollen, so verschieden nicht sein können.

Nicht nur, daß mit „Freier Software“ oder „Open Source“ exakt die gleichen Programme bezeichnet werden – auch ein Blick auf die Organisationen „Free Software Foundation“ und „Open Source Initiative“ bestätigt diesen Verdacht: Wichtige Akteure wie Lawrence Lessig und Eben Moglen sind in beiden Gruppierungen engagiert.[619] Daß Stallman und Raymond außerdem jeweils auf die gleichen sozialen Wurzeln, die Hacker-Gemeinschaft verweisen, wurde bereits in Kapitel 3.4.3.3 dargestellt.

Stallmans Behauptung, es handle sich um zwei Bewegungen, muss also möglicherweise einfach nur als Teil seines Kampfes um den Führungs- und Deutungsanspruch über die (eine) Bewegung verstanden werden – als Bemühen darum, die den Hackern immanente Abneigung gegen das „Forking“ eines Projekts – GNU/Linux – zu mobilisieren. Wie wenig Erfolg er damit hat, zeigt die Einhelligkeit, mit der der Begriff „Open Source“ von den Entwicklern freier Software unterstützt wurde und wird. Nach seinem ersten Scheitern mit der Distribution Debian mußte Stallman mit dieser erneuten Niederlage endgültig seinen Anspruch auf die Führung der GNU/Linux-Gemeinschaft begraben.

Ob er aber seine strategische Rolle gänzlich verloren hat, wie Raymond behauptet,[620] ist zu bezweifeln. Sein Einfluß auf freie Projekte[621] und auf die Haltung der Entwicklergemeinschaft in aktuellen Fragen muß nach wie vor als groß angesehen werden.[622] Auch wenn sie ihn als Führungskraft nicht akzeptieren, sind viele Entwickler wohl doch bereit, „RMS“ als „Gründervater“ Respekt und Gehör zu schenken.

Als Begriff jedoch hat „Open Source“ die Bezeichnung „Freie Software“ weitgehend abgelöst: 96 Prozent der rund 900.000 freien Entwickler, die ihre Projekte auf der Online-Plattform Sourceforge einstellen, verwendeten „Open Source“, stellte Raymond 2004

---

617 CLENDENNING, Alan: Activists Urge Free Open-Source Software. In: Yahoo! News. 29.01.2005, 17:32. Online unter: http://news.yahoo.com/news?tmpl=story&u=/ap/20050129/ap_on_hi_te/social_forum_4 [Aufgerufen am 31.01.2005].

618 So spricht HOLTGREWE vom „Innovationsmodell FS/OS“ und „FS/OS-EntwicklerInnen“. (HOLTGREWE, a.a.O., S. 340; S. 344) ZIMMERMANN verwendet die Begriffe „FS/OS-Akteuren“ sowie „FS/OS-Bewegung“ (ZIMMERMANN, a.a.O., S.361; S. 360).

619 Vgl. Zentrum für Rechtsfragen rund um Open Source gegründet. heise-online, 01.02.2005, 16:09 Uhr. http://www.heise.de/newsticker/meldung/55799 [Aufgerufen am 03.02.2005].

620 Dieser stellte bereits ein Jahr nach Einführung des Begriffs „Open Source“ fest, daß die gleiche Presse, die die Idee „freier Software“ zuvor als Hippie-Ideen belächelt hat, sich nun lobend über die Open-Source-Methode äußere. Er kommt zu dem Schluß: „OSI's tactics *work*. That's the easy part of the lesson. The hard part is that the FSF's tactics *don't work*, and never did.“ RAYMOND, Eric Steven: Shut Up And Show Them The Code. Veröffentlichung im Online-Magazin Linux Today vom 28.06.1999, 15 :51 Uhr. Archiviert unter: http://linuxtoday.com/news_story.php3?ltsn=1999-06-28-023-10-NW-SM [Aufgerufen am 05.02.2005].

621 Vgl. Kapitel 3.3.3 „Machtbegrenzung“ und Stallmans Rolle bei der Diskussion um das Projekt „XFree86“.

622 Vgl. Bill Gates sichert der EU-Kommission volle Kooperation zu. Meldung vom 02.02.2005, 14:50 Uhr. In: Heise Online, http://www.heise.de/newsticker/meldung/55853 [Aufgerufen am 3.01.2005].

fest.[623] Dabei wird der Begriff keineswegs nur technisch verstanden. Die Bedeutungen „offen“ und „Quelle“ eröffneten zahllose und – anders als bei „freier Software“ – auch erwünschte Interpretationsspielräume. Selbst außerhalb des Fachgebiets wurde „Open Source“ zum Modewort.[624]

Daß „Open Source“ sozusagen zum „Glasnost“ der IT-Branche geworden ist, mag auch daran liegen, daß der Begriff ein Leitbild der Hacker-Ethik aufgreift: Den freien Fluß von Information. In einer Welt aus offenem Code ist ein Hacker allmächtig.

Abbildung 30: „Open-Source“-Symbolik in dem 1999 erschienenen Film „Matrix“. Der Hacker „Neo“ sieht am Ende des Films die Welt als offenen Code – in der Abbildung den Hotelflur mit seinen drei Gegenspielern – und wird dadurch unbesiegbar.

---

[623] Vgl. LINUX Magazin: Open Source oder Free Software. Ausgabe 09/04. München: Linux New Media AG, 05.08.2004. S. 17 und RAYMOND, Eric Steven: Terminology Wars: A Web Content Analysis. Online veröffentlicht unter: http://www.catb.org/~esr/writings/terminology/ [Aufgerufen am 08.08.2004].

[624] Gebraucht wird Open Source etwa im Zusammenhang mit Harry-Potter-Fanliteratur: „Als eine Art literarischer Open-Source-Software dient Rowlings Hogwart-Saga dabei nur noch als grobes Bezugssystem, als Inspirationsquelle, Requisitenlager und Personalpool für eigene, immer feiner verästelte Potter-Abzweiger [...]“ HÄNTZSCHEL, Jörg: Dirty Harry. Deutsche Potter-Fans basteln an einer Kollektiv-Übersetzung. Süddeutsche Zeitung, 08.07.03. S. 13 (Feuilleton). Im Zusammenhang mit dem Humangenomprojekt taucht der Begriff ebenfalls auf: „Das Humangenomprojekt ist eine Open Source, alle haben den Zugang zu dieser Information.“ SCHWÄGERL, Christian: Der Mensch als Markt. Frankfurter Allgemeine Zeitung, 28.12.02, S. 29 (Feuilleton). Vgl. auch: SCHIRRMACHER, Frank: Die Patente der Fliege. Frankfurter Allgemeine Zeitung, 18.04.00, S. 49 (Feuilleton); HORN, Karen: Auf der Allmende des Wissens. Frankfurter Allgemeine Zeitung, 14.04.00, S. 13 (Leitartikel Wirtschaft) sowie SCHWÄGERL, Christian: Gute Träume vom manipulierten Leben. Frankfurter Allgemeine Zeitung, 28.02.03, S. 44 (Feuilleton).

3.6 Zusammenfassung

Die ursprüngliche Zielsetzung des GNU/Projekts, nicht nur ein freies Unix zu schaffen, sondern auch eine Erhaltung der Wertvorstellungen und die Art der Zusammenarbeit der frühen Hacker am MIT zu ermöglichen, ist erreicht worden – jedoch in einer völlig anderen Größenordnung, als das von Stallman 1983 abzusehen war.

Schneller als erwartet hatten sich zwei Technikleitbilder der IT-Branche erfüllt: Der „persönliche Computer", der in den 70ern als „Computer Power to the People!"[625] propagiert worden war, fand ab 1986 ausgerechnet als IBM-PC massenhafte Verbreitung. Und der Zugang zu Rechnern zu jeder Zeit und an jedem Ort, der als „computer that he can have continuously at his beck"[626], als „Rechner auf Wink" 1962 von John McCarthy ebenso gefordert worden war, wie 1990 durch Bill Gates („Information at your fingertips"[627]), fand in der massenhaften Verbreitung des Internets seine Verwirklichung.

Mit seiner Ankündigung eines „freien Unix für den PC" fand Linus Torvalds deswegen nicht nur eine hohe (und dank der rechtlichen Streitigkeiten der regulären Unix-Anbieter ungesättigte) Nachfrage von PC-Besitzern vor, sondern auch die Möglichkeit einer massenhaften Beteiligung von Freiwilligen an der Programmierung. Torvalds Wahl der GNU GPL als Lizenz garantierte diesen Freiwilligen eine faire Nutzung ihrer Beiträge; seine zurückhaltende Art der Projektleitung sorgte zudem dafür, daß sich die Freiwilligen auch innerhalb des Projekts, bei der Wahl ihres Aufgabenbereichs, frei fühlten und ermunterte sie so zu individuellen Beiträgen.

Die GNU GPL erwies sich für die Organisation der Gemeinschaft auch als strukturierendes Element: Sie stellte eine Machtbalance zwischen Projektleiter und Mitentwicklern her und sorgte durch das Damoklesschwert des „Forkings" für einen sparsamen Gebrauch von Autorität. Weil der Code nach der GPL von jedem frei verwendet und verbreitet werden kann, ist es grundsätzlich schwierig, Exklusivität herzustellen – mit seinen Ideen setzt sich langfristig nicht der durch, der Linus Torvalds am nächsten steht, sondern der, der die besseren Argumente hat. Die so entstandene Organisationsstruktur ähnelt dem Idealbild einer „Meritokratie", in der die Hierarchie ausschließlich durch die Reputation bestimmt wird, die sich die Organisationsmitglieder durch ihre Leistungen für die Gemeinschaft erworben haben.

Die Motive von Freiwilligen, sich nicht nur GNU/Linux auf dem PC zu installieren, sondern auch daran mitzuwerkeln, sind empirisch inzwischen weitgehend erforscht; hier spielen Eigennutz, Lernen und Altruismus eine Rolle, vor allem aber Spaß an der Tätigkeit des Programmierens. Weitgehend unerforscht sind jedoch die gemeinsamen Werte, die durch eine solche Gemeinschaft entstehen: die Organisationskultur. Hier zeigt sich gelegentlich ein pseudoreligiöses Selbstbild der Gemeinschaft, das mehr mit Kult als mit

---

625 NELSON, Theodor H.: Computer Lib/Dream Machines: New Freedom Through Computer Screens – a Minority Report. Chicago: Hugo's Book Service: 1974. Zitiert nach: MOODY, a.a.O., S.175.

626 McCARTHY, John: Time Sharing Computer Systems. Cambridge, Mass: MIT Press, 1962. Zitiert nach: LEVY, a.a.O., S.67.

627 GATES, William: Information at your fingertips 2005. Eröffnungsrede zur Messe „Comdex" am 14.11.1994. Online archiviert unter: http://www.microsoft.com/billgates/speeches/industry&tech/iayf2005.asp [Aufgerufen am 05.02.2005]. In der Rede heißt es, Microsoft verfolge dieses Ziel bereits seit 1990.

Kultur zu tun hat. Daß Filme wie „Star Wars“[628] als Wertematrix verwendet werden, könnte auch daran liegen, daß die Zielgruppe von Science Fiction (junge männliche Technologiebegeisterte) mit der Gemeinschaft identisch ist. Die schon früh auftretenden und häufigen Verweise auf Hacker-Traditionen und Werte dienen der GNU/Linux-Gemeinschaft als Selbstvergewisserung.

Die in Kapitel 1 beschriebenen Wertvorstellungen der MIT-Hacker stimmen mit denen der GNU/Linux-Gemeinschaft überein. Die Forderung nach freiem Zugang zu Computern entspricht der Forderung nach rechtlich unbedenklichem Zugang zum Quellcode. Das Copyleft-Prinzip in der GPL verwirklicht die Forderung nach der Freiheit von Information. Die Ablehnung von Autorität spiegelt sich in dem zurückhaltenden Managementstil Torvalds’ und letztlich sogar in der „Entmachtung“ Stallmans. Dem Ideal einer Meritokratie nähern sich beide Gemeinschaften gleichermaßen, wobei LEVYs Einschränkung gilt, daß dies keinen sozialen Motiven entspringt;[629] der Erwerb von Anerkennung ist nur durch „Hacken“ möglich und durch nichts anderes. Die Überzeugung, daß Programmieren Kunst sein kann, ist aus der GNU/Linux-Gemeinschaft ebenfalls häufig geäußert worden,[630] auch die Gefühle von Spaß bzw. „Flow“, die mit der Kreation von Code aus logischen Gedankengängen einhergehen können, verweisen darauf.

„Computers can change your life for the better“[631] nannte LEVY als letzte „Regel“ der MIT-Hacker. Schaut man sich unter den Anhängern von GNU/Linux um[632] so spiegelt sich der damalige technische Optimismus heute in ihrem Glauben, daß freie Software bzw. das Open-Source-Prinzip ganz allgemein das Leben und die Welt zum besseren verändern könnten.

Diese Verbindung von technischer und sozialer Utopie schlägt sich etwa darin nieder, daß die Free Software Foundation Europe einen Satz von Aurelius Augustinus als ihr Motto gewählt hat[633] – und für ihre Anhänger auf T-Shirts drucken läßt.[634] Sie zeigt sich darin, daß die „Open Source Definition“ ausdrücklich verbietet, daß Anwender von Open-Source-Software diskriminiert werden – kein Hersteller kann etwa verbieten, daß die von ihm freigegebene Open-Source-Software auch in Abtreibungskliniken verwendet wird.[635] Ein weiteres Beispiel zeigt sich bei der derzeit erfolgreichen GNU/Linux-Distribution „Ubuntu“: Sie hat sich nach dem südafrikanischen Ideal des Gemeinschaftsgeistes benannt; das Zulu-Wort Ubuntu liegt der Verfassung des Landes zugrunde und bedeutet so-

---

628 Außer „Star Wars“ wären z. B. zu nennen „Per Anhalter durch die Galaxis“, die „Futurama“-Zeichentrickfilme oder „Matrix“.

629 Siehe Abschnitt 1.3.4.

630 Vgl. TORVALDS: Just for…, a.a.O., S. 82 und RAYMOND, The Art..., a.a.O.

631 LEVY, S.45.

632 Oder auch in der Literatur darüber, als hervorragendes Beispiel dient etwa GRASSMUCK, a.a.O.

633 „Omnis enim res, quae dando non deficit, dum habetur et non datur, nondum habetur, quomodo habenda est.“ AUGUSTINUS: Doctrina Christiana. Unbekannter Ort. Versuch einer Übersetzung: „Denn jede Sache, die, wenn sie gegeben wird, nicht verloren wird, wird, wenn sie besessen und nicht gegeben wird, niemals so besessen, wie sie besessen werden sollte."

634 FSF EUROPE: Merchandise & Co. Webseite unter: http://www.germany.fsfeurope.org/order/order.de.html [Aufgerufen am 05.05.2005].

635 Vgl. PERENS, The Open..., a.a.O., S. 179. Aus gut gemeinten Gründen kommt es immer wieder zu Diskriminierung von Anwendern. So hatte die Universität von Kalifornien, Berkley, während der Zeit der Apartheid in Südafrika in einer ihrer Softwarelizenzen die Verwendung der Software durch die südafrikanische Polizei verboten.

viel wie „Ein Mensch ist ein Mensch durch andere Menschen“.[636] Auf der Homepage der Distribution heißt es: “The Ubuntu Linux distribution brings the spirit of Ubuntu to the software world.”[637]

Es geht nur um Software. Das muß bei all diesem Pathos und mancherlei Hybris immer wieder betont werden. Es geht nicht um Kommunismus.[638] GNU/Linux ist keine Möglichkeit, die Welt zu retten. Gegenteilige Äußerungen sind lediglich Ausdruck eines technischen Enthusiasmus, der den Berufsstand von Anfang an begleitet hat. Es wäre aber ungerecht, das soziale Verantwortungsgefühl, das aus den oben genannten Beispielen spricht, deswegen geringzuachten. „Linux hat das Soziale im Programmierer entdeckt“[639] stand vor einiger Zeit in der Zeitung. Diese Einschätzung ist falsch. Die Programmierer haben sich selbst als soziale Gruppe definiert, die ihre Mitglieder, ihre Arbeit und ihre Entwicklungsrichtung nach eigenen Werten beurteilt und selbstbestimmt gestaltet. In einer der kargsten und isoliertesten Umgebungen der Industriestaaten, vor einem Rechner, ist eine Arbeitsethik entstanden, deren Produkte die Branche mehr und mehr umwälzen und die sich in grundlegenden Wertvorstellungen früher Christen und sogar der Zulu-Kultur wiederfindet.

---

[636] WIKIPEDIA (englisch): Ubuntu. Veröffentlicht unter: http://en.wikipedia.org/wiki/Ubuntu [Aufgerufen am 02.02.2005]. Der Artikel verweist auf: TUTU, Desmond Mpilo: No Future Without Forgiveness. New York: Image/Doubleday, 1999, S. 31.

[637] Welcome to Ubuntu. Homepage der GNU/Linux-Distribution unter: http://www.ubuntu.com/ [Aufgerufen am 06.02.2005].

[638] Ironischerweise wird dieser Vorwurf ausgerechnet von dem Monopolisten erhoben, dem mit GNU/Linux nun erstmals seit Jahrzehnten ein ernstzunehmender Wettbewerber herangewachsen ist: „Microsoft-Gründer Bill Gates hat sich unterdessen auf der CES gegen jene gewandt, die eine Lockerung bestehender Urheberrechtsgesetze fordern. Seiner Meinung nach ist geistiges Eigentum der Antrieb, um die Produkte von morgen zu entwickeln. Wer die Copyright-Gesetze abschwächen wolle, gehöre einer ‘neuen Art von Kommunisten’ an, die den ‘American Way’ gefährdeten.“ Softwareverband fordert Änderung der US-Copyright-Gesetze. heise Newsticker vom 07.01.2005, 15:35 Uhr. Online veröffentlicht unter: http://www.heise.de/newsticker/meldung/54895 [Aufgerufen am 07.01.2005].

[639] BERGER, Jürgen: Kalt ist der Datenhauch. Generation Gates: Douglas Couplands Roman „Microsklaven“ wurde für das Theater Basel adaptiert und dort uraufgeführt. Süddeutsche Zeitung, 28.11.03. S. 16 (Feuilleton).

Schlußbemerkung

„Linux is now mainstream“[640] hat das Marktforschungsunternehmen IDC Anfang 2005 festgestellt, es sei kein Nischen-Betriebssystem mehr. Den Umsatz mit GNU/Linux-Produkten veranschlagen die Experten auf derzeit 15 Milliarden US-Dollar pro Jahr. Über die Zahl der Installationen weltweit gibt es nur Schätzungen: IDC geht in diesem Jahr von gut 14 Millionen Rechnern mit diesem Betriebssystem aus, rechnet aber mit hohen Steigerungsraten (44,3 Prozent pro Jahr).[641] Andere nennen 18[642] oder gar 30[643] Millionen Installationen. Hinzu kommt der Erfolg weiterer freier Software wie des Webservers „Apache“, der seit Jahren Marktanteile gewinnt,[644] oder des neuen Lieblings der Medien, des freien Browsers „Firefox“.

Angesichts dieses Erfolges freier Software und damit der Hacker-Ethik soll im folgenden überlegt werden, ob und wie Medienunternehmen und Journalisten davon lernen und profitieren können. Es handelt sich dabei um einen kurzen Überschlag der vorhandenen Möglichkeiten; Anspruch auf Vollständigkeit wird nicht erhoben.

Der Einsatz freier Software in Medien

Freie Software ist in einem Siegeszug begriffen, dem sich die meisten Medienunternehmen in Deutschland nicht verschließen. Auch sie nutzen in irgendeiner Form freie Software. Am leichtesten zu belegen ist das durch eine Untersuchung des Webauftritts: Die über den Dienst Netcraft ermittelten Serverkennungen der Webseiten von *ARD*, *ZDF*, *RTL*, *SAT1*, *n-tv*, *Bild-Zeitung*, *Süddeutsche Zeitung*, *Frankfurter Rundschau* und *taz* zeigen, daß diese entweder GNU/Linux als Betriebssystem einsetzen oder „Apache“ als Server.[645] Natürlich ist diese Ermittlungsweise recht grob; hinter einer Internet-Adresse verbirgt sich üblicherweise eine Vielzahl von Serversystemen.

An den eigentlichen Arbeitsplätzen der Print-, Rundfunk- und Onlineredakteure dürften dagegen schon wegen des sonst notwendigen Schulungsbedarfs Windows- oder MacOS-Oberflächen dominieren. Einzige bekannte Ausnahme ist die *taz*, bei der auch das Redaktionssystem unter Linux läuft – aus finanziellen, aber auch aus ideologischen Gründen.[646]

---

640 IDC Software Consulting: The Linux Marketplace – Moving From Niche to Mainstream. December 14, 2004. Studie im Auftrag des Open Source Development Labs (OSDL). Online veröffentlicht unter: http://www.osdl.org/docs/linux_market_overview.pdf [Aufgerufen am 16.12.2004], S. 15.

641 Vgl. ebd.

642 Vgl. ALVESTRAND, Harald: The Linux Counter. Online veröffentlicht unter: http://counter.li.org/ [Aufgerufen am 06.02.2005].

643 GRASSMUCK, a.a.O., S. 229.

644 Vgl. February 2005 Web Server Survey Finds 40 Million Sites on Apache. Artikel auf der Webseite der Firma Netcraft vom 01.02.2005 01:12 Uhr. Online veröffentlicht unter: http://news.netcraft.com/archives/2005/02/01/february_2005_web_server_survey_finds_40_million_sites_on_apache.html
[Aufgerufen am 05.02.2005].

645 Ermittelt durch jeweilige Abfragen bei Netcraft.com im Feld: „What's that site running?“. Von den großen überregionalen Zeitungen setzt allein die FAZ auf proprietäre Software und ein Gespann aus Microsoft Windows 2000 und IIS 5.0.

646 Vgl. die Biographie von Ralf KLEVER unter: http://www.taz.de/pt/.etc/nf/kg/teile
[Aufgerufen am 07.08.2004].

Bei der angenommenen Windows-Dominanz lohnt jedoch ein Blick hinter die Fenster: Denn die browserbasierten Abfrage- und Content-Management-Systeme, etwa für Archiv-, Artikel- oder Bildverwaltung greifen im Hintergrund auf Systeme zu, bei denen oft freie Software eingesetzt wird: So arbeiten die Online-Redakteure im ZDF zwar mit dem Internet Explorer an Windows-Clients, die technische Basis des Digital-Angebots beim ZDF besteht dagegen, wie AVERKAMP berichtet,[647] aus einem Cluster von Suse-Linux-Rechnern. Insgesamt werden im Online- und Digitalbereich des Senders rund 60 GNU/Linux-Server eingesetzt. Auch das Fernsehprogramm des ZDF läuft auf Freier Software: So dient Red Hat Linux als Server-Betriebssystem für das *„Digitale Produktionssystem Aktuelles“*, eine Software der Firma Avid, mit der alle „heute“-Nachrichten gesendet werden.

Wie freie Software auch kleinere Nischen in Medienunternehmen besetzt, zeigt sich bei der ARD Tagesschau:[648] Deren Online-Mitarbeiter benötigten eine Software, um die Nachrichten aus dem Fernsehen auf dem Computer aufzuzeichnen, in Einzelbeiträge zu zerschneiden, mit den nötigen Textanmerkungen zu versehen und sie dann schließlich ins Internet zu stellen. Statt für teure Videobearbeitungsprogramme entschied sich der zuständige Projektleiter für die freie Software „Kino“[649]. In den folgenden Monaten setzte sich die ARD mit dem „Kino“-Projektleiter, einem amerikanischen Familienvater aus Cincinnati im Bundesstaat Ohio, in Verbindung und gab Anpassungen in Auftrag.[650] Seit Mai 2004 ist das System im Einsatz. Der Verantwortliche der ARD, Froehling, erklärt dazu, ihn habe „vor allem die Anpaßbarkeit der Open-Source-Software überzeugt“[651]. Und in den „Release-Notes“, sozusagen dem Logbuch der Software, heißt es: „Dank an Tagesschau.de für die Unterstützung vieler neuer Features.“[652]

Die ARD als Entwickler von freier Software. Man möchte die Frage stellen, warum die Eigenentwicklungen der öffentlich-rechtlichen IT-Abteilungen nicht grundsätzlich nach der GPL freigegeben werden. Es ist nicht ersichtlich, warum das, was hier mit öffentlichen Mitteln bezahlt wurde, nicht auch auf direktem Wege wieder der Öffentlichkeit zugute kommen sollte. Schließlich konkurriert man mit den privaten Sendern nicht um Software, sondern mit Inhalten um Zuschauer. Durch eine Beteiligung oder gar die Leitung von Open-Source-Projekten könnten die Rundfunkanstalten möglicherweise sehr viel weitergehender an technischen Innovationen teilhaben – und sie vielleicht sogar steuern – als es durch die politisch motivierten technischen Zielsetzungen der Intendanz möglich ist, die von den Zuschauern oder den Käufern von Fernsehgeräten dann oft nicht angenommen werden (HDTV, EPG).

---

647 AVERKAMP, Peter: Linux-Cluster beim ZDF. Beim Zweiten sendet man ausfallsicher. Linux Enterprise 7/8.04. Frankfurt am Main: Software & Support Verlag GmbH: 09.06.2004, S. 39-42.

648 FROMMEL, Oliver: Tagesschau weltweit mit Linux. Das DV-Schnittprogramm Kino bei Tagesschau.de. Linux Magazin, Ausgabe 08/04. München: Linux New Media AG, 08.07.2004. S. 79.

649 Website: http://kino.schirmacher.de/ [Aufgerufen am 07.08.2004].

650 Davon berichtet etwa der Kino-Entwickler Dan Dennedy auf seiner Webseite: „Collaborated with a developer in Germany for popular German news website tagesschau.de. The project paid for enhancements to Kino, my open source non-linear DV video editor that has been under development since January, 2000. In particular, the project required custom integration of the video editor with their content management system to automate the process of creating streaming media for news articles. This project is implemented in C++, XML, and shell scripts for the GNU/Linux platform.“ http://www.dennedy.org/resume.html [Aufgerufen am 07.08.2004].

651 FROMMEL, a.a.O.

652 Ebd.

Die Freigabe von journalistischen Produkten

Nicht nur Software-Hersteller, auch Medienunternehmen verkaufen Informationen – und diese sind inzwischen ebenfalls nicht mehr ausschließlich an physikalische Träger gebunden, wie Zeitungen, Bücher oder Bänder. „Wir dachten bisher vielleicht, die New York Times verkaufe Zeitungen, doch der Cyberspace lehrt uns, daß sie Editionsdienstleistungen verkauft, die in diesem Falle auf Papier erscheint,“[653] schreibt der amerikanische Rechtsprofessor Lawrence LESSIG.

Angesichts der Gleichartigkeit von Programmcode und Medienprodukten im Internet liegt der Gedanke nahe, das Modell der GNU GPL auch auf andere Werke, etwa auf journalistische Produkte, zu übertragen. Entsprechende Lizenzen wurden in den vergangenen Jahren vorgelegt: So unterliegen etwa die Inhalte der Online-Enzyklopädie „Wikipedia“ und ihres Nachrichtendienstes „Wikinews“ der „GNU Free Documentation License“, einer Urheberrechtsbestimmung, die von Stallman für die Freigabe von Computerhandbüchern gedacht war. Alle Inhalte von Wikipedia dürfen damit frei verändert, verbreitet und verkauft werden. Außerdem sind in Deutschland erstmals Bücher unter der „Creative Commons Lizenz“ erschienen,[654] sie ermöglicht auch restriktivere Regelungen, die keine kommerzielle Weiterverarbeitung und -verbreitung des heruntergeladenen Lesestoffs erlauben. Um die Gunst von Autoren, Künstlern und anderen „Content-Providern“ im wörtlichen Sinne werben auf diese Art zahllose weitere Lizenzen, etwa die „Free Art License“, „Open Content“, „Science Commons“ oder die „Artistic License“.

Andere führen bereits eine Debatte, ob Copyright im Internetzeitalter überhaupt noch eine Rolle spielen sollte: GRASSMUCK sieht hier Chancen auf die Schaffung einer „Wissensallmende“, der „Abkopplung der Ideenwirtschaft von der normalen Güterwirtschaft“[655], bei der Wissen – wie in der öffentlichen Forschung – zum Gemeingut wird: „Das überlieferte Versprechen des Internet ist es, das ‚Große Wissensreservoir’ und der ‚Große Dialog’ der Menschheit’ zu werden.“[656] Der Berliner Mediengeschichtler Friedrich KITTLER hält das „Zusammenschmeißen eines alten, aus der Goethezeit stammenden Copyright- und Eigentumsrechts an geistigen Produkten mit den neuen Entdeckungen der digitalen Maschinen“[657] schlicht für eine Katastrophe.

Auf der anderen Seite wehren sich namhafte Autoren gegen das Kopieren ihrer Werke: So meint Vilem FLUSSER: „Kopieren macht alle Autorität und alle Autoren überflüssig und stellt daher die schöpferische Begeisterung in Frage.“[658] Die Musik-Verwertungs-

---

653 LESSIG, a.a.O., S.302.

654 WOS3: Telepolis-Bücher unter Creative-Commons-Lizenz als freier Download. Heise Online vom 11.06.2004, 17:50 Uhr. Veröffentlicht unter: http://www.heise.de/newsticker/meldung/48179 [Aufgerufen am 05.02.2005].

655 GRASSMUCK,a.a.O., S. 178. GRASSMUCK verweist in diesem Zusammenhang auf SPINNER, Helmut F.: Die Wissensordnung. Ein Leitkonzept für die dritte Grundordnung des Informationszeitalters. Opladen: Leske+Budrich, 1994, S. 91.

656 Ebd., S. 400.

657 MATTHEISS, Uwe: Krieg der Kopierer. Das Urheberrecht in Zeiten weltumspannender Informationsnetze. Süddeutsche Zeitung, 28.09.98. S.13 (Feuilleton).

658 Zitiert nach ebd.

gesellschaft Gema verbietet ihren Mitgliedern sogar, ihre Werke unter eine freie Lizenz zu stellen, da sie sich das Alleinvertretungsrecht ihrer Künstler vorbehält.[659]

Diejenigen, die am lautesten die Befreiung der Information fordern, sind nicht immer deren Schöpfer, wie ein Beispiel von „attac", der Organisation der Globalisierungsgegner, zeigt: So forderte die „AG Wissensallmende" den Dachverein im Frühjahr 2004 auf, folgendem Antrag zuzustimmen:

> „Alle Publikationen (Bücher, Broschüren, CDs, etc.) im Namen von attac sollen zugleich auch vollständig und unentgeltlich auf der Website von attac der Öffentlichkeit zugänglich gemacht werden. Dies gilt insbesondere für die attac-Basistexte. [...] Die Inhalte von Musik-CDs, die attac herausgibt, sollen ebenfalls in digitaler Form (Ogg-Vorbis, MP3) im Internet veröffentlicht werden, soweit dadurch keine GEMA-Gebühren zu zahlen sind. Alle komplexere Software, die für ATTAC erstellt wird oder im Namen von ATTAC verfügbar gemacht wird, sollte unter die GPL oder einer zur GPL kompatiblen Software Lizenz gestellt werden."[660]

Drei Monate später hieß es kleinlaut:

> „Am 8. und 9. Mai 2004 veranstaltete unsere AG ein Seminar zur Vorstellung und über Alternativen zu Softwarepatenten und immer schärferem Urheberrecht. Unser Antrag zur attac-Veröffentlichungspolitik wurde erst einmal abgelehnt. Unsere Positionen müssen also auch innerhalb Attacs noch besser vertreten werden."[661]

Auch die Hacker selbst sind von der Forderung nach der Freigabe aller geistigen Produkte nicht überzeugt. Stallman sieht dafür auch keinerlei Notwendigkeit:

> "As a general rule, I don't believe that it is essential for people to have permission to modify all sorts of articles and books. The issues for writings are not necessarily the same as those for software. For example, I don't think you or I are obliged to give permission to modify articles like this one, which describes our actions and our views."[662]

LESSIG spricht in diesem Zusammenhang von Extrempositionen, die in der Diskussion eingenommen werden: Vor allem der Medienindustrie spiele die Forderung nach einem Ende des Copyrights immer wieder in die Hände; sie stelle die Diskussion grundsätzlich so dar, als ob es um einen Krieg zwischen amerikanischen und kommunistischen Werten ginge.[663] Unter dem Deckmantel dieses Krieges gegen die ‚Raubkopierer im Internet' hätten Musik- und Filmindustrie den rechtlichen Schutz ihrer Produkte auf bis zu 150 Jahre verlängert.[664] Er sieht darin eine Verletzung der Balance zwischen öffentlichem und privatem Interesse:

> "We live in a world with parks as well as private property, with public roads as well as private drives. We in this world understand the importance of balance between private and public. It would be silly to sell the sidewalks; it would be crazy to nationalize GM

---

659 KREMPL, Stefan: WOS3: Creative Commons als Geheimwaffe der Künstler im Copyright-Krieg. heise-online, 12.06.2004, 11:12 Uhr. http://www.heise.de/newsticker/meldung/48184 Abgerufen am 06.08.2004].

660 http://www.attac.de/wissensallmende/shareddocs/wafi-antrag01.php [Aufgerufen am 12.08.04].

661 http://www.attac.de/wissensallmende/ [Aufgerufen am 12.08.04].

662 STALLMAN, Richard Matthew: Free Software Needs Free Documentation. In (ders.): Free Software, Free Society. Selected Essays. Boston: GNU-Press, 2002, S. 67.

663 LESSIG: The Future..., a.a.O., S. XVI.

664 Ebd., S. 251.

> [General Motors]. Yet when it comes to cyberspace, and in particular, to the laws that regulate cyberspace, we are increasingly forgetting this lesson of balance."[665]

LESSIG schlägt als Alternative für den Umgang der Informationsgesellschaft mit Information eine Abstufung der Schutzfristen vor:[666] Heutzutage seien viele Werke automatisch jahrzehntelang urheberrechtlich geschützt, obwohl noch nicht einmal ihre Autoren daran noch ein Interesse hätten. Er fordert statt dessen – bei Beibehaltung der bestehenden Fristen – daß das Copyright alle fünf Jahre gegen Gebühr verlängert werden müsse. So könnten der Autor, sein Verlag oder seine Nachkommen jeweils neu entscheiden, ob das Werk noch den strengen Schutz wert sei oder zum Allgemeingut werden könne.

Auch wenn dieses Modell auf das amerikanische Verständnis von Copyright zugeschnitten ist, könnte eine ähnliche Vorgehensweise auch für den deutschen öffentlich-rechtlichen Rundfunk interessant sein – etwa bezüglich der umfangreichen Archive journalistischer Beiträge, die aufgrund von Rechtsunsicherheiten sogar von den Anstalten selbst nur zu geringem Teil genutzt werden können. Hier wäre eine vertragliche Regelung mit den Autoren, etwa daß Beiträge nach einer gewissen Schutzfrist (z.B. fünf Jahre zur Erhaltung von Exklusivität) unter eine freie Lizenz fallen, möglicherweise in aller Interesse: Für die Anstalt entfiele die Speicherung und langwierige Prüfung von Altverträgen, die Autoren hätten die Gewißheit, daß ihr Beitrag nicht nach ein paar Jahren nutzlos in einem „schwarzen Loch" verschwindet und die Allgemeinheit, die die Aufnahmen ja sowieso bezahlt hat, käme in den Besitz von Zeitdokumenten, die zur Lehre, aber auch zur Konstruktion „frischer" Werke frei verwendet werden könnten.[667]

Open-Source-Journalismus

Am vielversprechendsten und interessantesten scheint derzeit aber die Umsetzung der *Methoden* der Hacker auf andere Bereiche: So hat die angesehene Harvard Law School seit 1998 das Projekt „Open Law" gestartet; dabei werden nicht nur – wie üblich – die Gerichtsprotokolle mit Aussagen, Plädoyers und Richterspruch veröffentlicht, sondern die Anwälte diskutieren ihre Argumente in konkreten juristischen Fällen vorab in einem offenen Webforum. So wollen sie ihre optimale Vorgehensweise herausfinden. An einem derartigen Fall sind inzwischen 50 freiwillige Juristen beteiligt.[668]

Auch im Journalismus gibt es Ansätze, die das gemeinschaftliche Entwicklungsprinzip freier Software übernehmen. So trat am 08. Oktober 1999 erstmals der Begriff „Open-Source-Journalismus" in Erscheinung, in einem Artikel, den der amerikanische Essyist Andrew LEONARD für die Online-Zeitung „salon.com" verfaßte. Ausgelöst worden war LEONARDs Artikel durch einen ungewöhnlichen Vorfall bei der amerikanischen Waffenzeitschrift „Jane's Intelligence Review". Jane's hatte einen Artikel über Cyberterrorismus in Auftrag gegeben, den fertigen Text einige Tage vorab dem Online-Magazin

---

665 Ebd, S. XIXf.

666 Ebd., S 251f.

667 Eine solche Freigabe würde sich natürlich auch auf die Vertragsgestaltung mit freien Zulieferern und z.B. der European Broadcast Union (EBU) auswirken. Hier sitzen aber die Anstalten selbst am längeren Hebel.

668 WIKIPEDIA (englisch): Openlaw. Online veröffentlicht unter: http://en.wikipedia.org/wiki/Open_Law_project [Aufgerufen am 06.02.2005].

„Slashdot“ zugespielt und die Leser dieses Magazins – Computerfachleute und Programmierer – um Anregungen und Kritik gebeten.

Diese Kritik war so verheerend, daß der Herausgeber von Jane’s, Jonathan Ingles-le Nobel im Slashdot-Forum ankündigte, den Abdruck des Artikels vorerst auf Eis zu legen. „’Wenn man um Feedback bittet, bekommt man Feedback,’ schrieb Nobel, ‚und weil ungefähr 99 Prozent der Schreiber den Artikel in Bausch und Bogen verdammten, noch dazu Dinge sagten wie ‚wir hätten Besseres von Jane’s erwartet’, habe ich den Autoren verständigt, daß wir den Artikel nicht bringen.’“[669] Statt dessen kündigte Nobel an, die Kommentare und Argumente der Slashdot-Leser in einem neuen, verbesserten Artikel zum Thema zu berücksichtigen und den Lesern, die er dabei wortwörtlich zitieren würde, ein Honorar zu zahlen.

Während einige[670] daraufhin von einer Art Zensur durch die Netzgemeinde sprachen, verglich LEONARD diesen Vorgang mit der Open-Source-Arbeitsweise: „Genau wie Open Source Programmierer die Beta-Version eines Programms voller Fehler kritisieren würden, genau so haben auch die Slashdot-Leser auf die erste Version von Jane’s journalistischem Angebot reagiert – und das Upgrade wird nun offensichtlich bald folgen.“[671] Ob dies ein Modell für die Zukunft sei, bleibe aber ungewiß, schreibt LEONARD, die kurzen Deadlines der Computerjournalisten ließen gewöhnlich kaum Zeit für einen solchen „Beta-Test“. „Dennoch ist dieses Prinzip einen langen, genauen Blick wert. Es gibt eine immense Menge Expertenwissen draußen im Netz.“[672]

Was LEONARD vor sechs Jahren noch als skurrile und einmalige Aktion eines Verlags darstellte, wird inzwischen auch vor dem Hintergrund neuer journalistischer Formen wie elektronischer Tagebücher (Weblogs) oder gemeinschaftlich verfaßter Nachrichtenportale (z.B. Wikinews[673]) ernsthaft diskutiert.[674] Dabei geht es auch um Qualität: So behauptet etwa die erfolgreiche[675] Online-Enzyklopädie Wikipedia, daß sie eine objektivere Darstellung von Konflikten bieten könne, als herkömmliche Medien. Denn unter der Maßgabe, einen neutralen Standpunkt einzunehmen, können alle Parteien an der Darstellung mitschreiben:

> “Wikipedia’s neutral point of view policy makes it an excellent place to gain a quick understanding of controversial topics. Want a good overview of the Arab-Israeli conflict but only got 10 minutes to spare? Wondering what all the fuss is about in Kashmir or what the pro/con arguments are about stem cell research? Wikipedia is a great place to start.”[676]

---

669 LEONARD, Andrew: Open-source journalism. In salon.com, 8.11.1999, http://www.salon.com/tech/log/1999/10/08/geek_journalism [Aufgerufen am 10.08.04]. (Übersetzung: U.H.)

670 Darunter der Slashdot-Kolumnist Robert X. Cringely. Ebd.

671 Ebd.

672 Ebd.

673 Vgl. Wikinews, Onlineangebot unter: http://en.wikinews.org/wiki/Main_Page.

674 Vgl. GILLMOR, Dan: We the Media. Grassroots Journalism by the People, for the People. Sebastopol: O'Reilly, 2004.

675 Wikipedia verzeichnet derzeit rund 1000 Aufrufe pro Sekunde. Vgl. Wikipedia soll schneller werden. Linux-User, Ausgabe 02/05. München: Linux New Media AG. S. 8.

676 WIKIPEDIA (englisch): Wikipedia: Why Wikipedia is so great. Veröffentlicht unter: http://en.wikipedia.org/wiki/Wikipedia:Why_Wikipedia_is_so_great [Aufgerufen am 06.02.2005].

Viele Informationen der vorliegenden Arbeit stammen aus *Wikipedia*-Artikeln – die Qualität der Artikel im Computerbereich ist möglicherweise auch deshalb so überzeugend, weil hier eine hohe gemeinsame Kompetenz der Wikipedia-Nutzer[677] besteht. *Wikinews* dagegen befindet sich zum Zeitpunkt der Entstehung dieser Arbeit in einem unfertigen Stadium. Wagt man dennoch eine flüchtige Untersuchung, so fällt auf, daß die Qualität von Wikinews den Ansprüchen an ein Nachrichtenmedium nicht genügen: Die Artikel sind nicht aktuell, ihre Auswahl ist zudem nicht nachvollziehbar: So fehlten am 07.02.2005 in Wikinews die beiden Aufmacherthemen von CNN.com völlig (Waffenstillstand in Nahost angekündigt; 28 Tote bei Anschlag im Irak); der eigene Aufmacher war dagegen zwei Tage alt. Von den gerade mal zehn „News des Tages“ spiegelten zwei Computerthemen und eine behandelte das chinesische Neujahrsfest in Bukarest.

Wie in der vorliegenden Arbeit beschrieben, ist es ein wesentlicher Punkt bei der Organisation der Arbeit am Linux-Kernel oder auch an der Debian-Distribution, daß Zuständigkeiten klar zugeordnet werden: Es bilden sich Experten heraus, die für ein bestimmtes Fachgebiet zuständig sind. Umgekehrt wird die Arbeit so untergliedert, daß möglichst kleine, abgegrenzte Aufgabenbereiche entstehen. Bei einer Enzyklopädie ist dies ebenfalls gegeben: Experten können sich hier ihren eigenen Themengebieten, d.h. einzelnen Artikeln widmen.

Dieses Prinzip funktioniert im Nachrichtenjournalismus nicht. Hier ist jeder ein (wenn auch oberflächlicher) Experte für alles; Spezialwissen wird begrüßt, kommt aber nicht oft zur Geltung. Es bilden sich gemeinsame Wertmaßstäbe heraus, nach denen die Relevanz von Meldungen eingeschätzt wird. Ob international und kulturell verteilte Freiwillige, ohne Druck zur Aktualität und weitgehend ohne die Recherche bei Primärquellen jemals ein ähnlich hohes Qualitätsniveau erreichen wie etwa CNN.com, bleibt mit Spannung abzuwarten.

Das Prinzip des „Open-Source-Journalismus“ dagegen, ein Thema in Interaktion mit den Zuschauern bzw. den Lesern – etwa in Newsgroups – zu entwickeln und sich dort auch der Kritik zu stellen, könnte durchaus ein interessanter Ansatzpunkt für die eigene Arbeit sein. Dieses Prinzip würde möglicherweise auch dafür sorgen, daß Reputation nicht mehr vorwiegend unter den Kollegen erworben wird, sondern dort, wo es viel wichtiger ist: Bei den Lesern.

---

[677] Da jedermann ohne Anmeldung Wiki-Seiten einfach nur aus dem Webbrowser heraus beschreiben kann, sind die Nutzer zumindest theoretisch identisch mit den Autoren.

Bibliographie

ALBERT, Phil: A Consumer's Review of the General Public License. In: Linux Insider, 20.07.2004 5:45 AM PT. Online veröffentlicht unter: http://www.linuxinsider.com/story/35193.html [Aufgerufen am 04.12.2004].

ALVESTRAND, Harald: The Linux Counter. Online veröffentlicht unter: http://counter.li.org/ [Aufgerufen am 06.02.2005].

ANDRESEN, Fred; WOLF, Ulrich: Tux inside. Biotop für Pinguine: Joschka Fischers Auswärtiges Amt. Linux-Magazin, Ausgabe 05/2003. München: Linux New Media AG, S. 72-74.

AVERKAMP, Peter: Linux-Cluster beim ZDF. Beim Zweiten sendet man ausfallsicher. Linux Enterprise 7/8.04. Frankfurt am Main: Software & Support Verlag GmbH: 09.06.2004, S. 39-42.

BARLOW, John Perry: A Declaration of the Independence of Cyberspace. Online veröffentlicht unter: http://www.eff.org/~barlow/Declaration-Final.html [Aufgerufen am 25.10.2004].

BARLOW, John Perry: The Economy of Ideas. In: Wired, Ausgabe 2/03, März 1994. http://www.wired.com/wired/archive/2.03/economy.ideas.html Aufgerufen am 15.06.2004.

BAUER, Elisabeth; FROMMEL, Oliver: Auf Freiheit programmiert. Konferenz Wizards of OS. Linux Magazin, Ausgabe 08/04. München: Linux New Media AG, 08.07.2004, S. 82f.

BAUMGÄRTEL, Tilman: Am Anfang war alle Software frei. Microsoft, Linux und die Rache der Hacker. In: ROESLER, Alexander; STIEGLER, Bernd (Hrsg.): Microsoft. Medien, Macht. Monopol. Frankfurt a. M.: Suhrkamp, 2002, S. 103-129.

BELLIS, Mary: Inventors of the Modern Computer. Online veröffentlicht unter: http://inventors.about.com/library/weekly/aa090198.htm [Aufgerufen am 06.09.2004].

BERGER, Jürgen: Kalt ist der Datenhauch. Generation Gates: Douglas Couplands Roman „Microsklaven" wurde für das Theater Basel adaptiert und dort uraufgeführt. Süddeutsche Zeitung, 28.11.03. S. 16 (Feuilleton).

BERNERS-LEE, Tim: Weaving the Web. The Past, Present and Future of the World Wide Web by its Inventor. London, New York: Texere, 2000.

BIELICKI, Jan: Rote Äpfel und ein Fahrrad. Monica Lochner-Fischer (SPD) wirbt mit Obst und Linux – Ludwig Spaenle (CSU) lockt mit lokalen Themen. Süddeutsche Zeitung, 10.09.03, S. 47 (München).

BORCHERS, Detlef: ‚Open Source' – das Zauberwort des Jahres 1998. Wenn ein kleiner Drache Feuer spuckt. Warum hochwertige Software nichts kosten muß und dennoch große Geschäfte verspricht. Süddeutsche Zeitung, 29.12.98. S.10 (Computer).

BORCHERS, Detlef: Trau keinem unter www. So wichtig Vertrauen in der Online-Welt ist, so hart muss man es sich erarbeiten. Süddeutsche Zeitung, 12.08.03, S. 17 (Wissenschaft).

BORCHERS, Detlef: Verteiltes Nachdenken. 20 Jahre kritische Informatik. In: c't Magazin für Computertechnik. Ausgabe 22/2004. Hannover: heise, 18.10.2004, S. 110-111.

BRAND, Stewart: The Media Lab: Inventing the Future at MIT. New York: Viking Penguin, 1987.

BRAUER, Wilfried: Tätigkeitsbericht 1991. Theoretische Informatik und Grundlagen der KI. München: Institut für Informatik, Technische Universität München, 1991. Online veröffentlicht unter:
http://wind.in.tum.de/taetigkeitsberichte/taet91/taet91.html
[Aufgerufen am 08.01.2005].

BRUCHERSEIFER, Eva: Die KDE-Entwicklergemeinde – wer ist das? In: GEHRING, Robert A.; LUTTERBECK, Bernd (Hrsg.): Open Source Jahrbuch 2004. Zwischen Softwareentwicklung und Gesellschaftsmodell. Berlin: Lehmanns Media, 2004, S. 65-82.

BRÜGGE, Bernd; HARHOFF, Dietmar; PICOT, Arnold; CREIGHTON, Oliver; FIEDLER, Marina; HENKEL, Joachim: Open-Source-Software. Eine ökonomische und technische Analyse. Berlin: Springer, 2004.

BUSHNELL, Thomas: What's up with the GFDL? Mail von Montag, 17 Nov 2003 11:33:16 -0800 an die Mailingliste gnu-prog-discuss@gnu.org. Veröffentlicht unter:
http://lists.softwarelibero.it/pipermail/discussioni/2003-November/008465.html [Aufgerufen am 12.12.2004].

Business Software Alliance: Studie: IDC Pirateriezahlen. Veröffentlicht unter:
http://www.bsa.org/germany/piraterie/piraterie.cfm
[Aufgerufen am 11.12.2004].

c't Magazin für Computertechnik: „Unbemannte Flugzeuge per Sprache steuern". Ausgabe 24/2004. Hannover: heise, 15.11.2004, S. 54.

c't Magazin für Computertechnik: Verfügung wegen GPL-Verletzung. Ausgabe 10/2004. Hannover: heise, 03.05.2004, S. 44.

CHALLET, Damien; LE DU, Yann: Closed source versus open source in a model of software bug dynamics. Online veröffentlicht unter: http://arxiv.org/abs/cond-mat/0306511 [Aufgerufen am 02.02.2005].

CLENDENNING, Alan: Activists Urge Free Open-Source Software. In: Yahoo! News. 29.01.2005., 05:32 PM. Online veröffentlicht unter. http://news.yahoo.com/news?tmpl=story&u=/ap/20050129/ap_on_hi_te/social_forum_4 [Aufgerufen am 31.01.2005].

COX, Alan: Maintainers And Source Submission Procedures. Kernel-Patch 1.3.68 vom 22.02.1996. Online verfügbar unter anderem bei: http://sunsite.unc.edu/pub/Linux/kernel/v1.3/patch-1.3.68.gz [Aufgerufen am 18.01.2005]. „P:“ steht für „Person“, „S:“ für „Status“.

CUNNINGHAM, Donna; REGAN; Patrick: Ritchie and Thompson Receive National Medal of Technology from President Clinton. Pressemitteilung der Fa. LUCENT Technologies / Bell Labs Innovations. Online unter: http://www.bell-labs.com/news/1999/april/28/1.html [Aufgerufen am 10.10.2005].

DEMPSEY, Bert J.; WEISS, Debra; JONES, Paul; GREENBERG, Jane: A Quantitative Profile of a Community of Open Source Linux Developers. October 1999. Online veröffentlicht unter: http://www.ibiblio.org/osrt/develpro.html. [Aufgerufen am 18.01.2005].

DIBONA, Chris; OCKMAN, Sam; STONE, Mark (Hrsg.): OpenSources. Voices from the Open Source Revolution. Sebastopol: O’Reilly, 1999.

DIEDRICH, Oliver: Happy Birthday, Tux! Zehn Jahre Freies Betriebssystem. In: c’t Magazin für Computertechnik. Ausgabe 19/2001. Hannover: heise, 10.09.2001, S. 162-169. Online veröffentlicht unter: http://www.heise.de/ct/01/19/162/default.shtml [Aufgerufen am 18.01.2005].

DIERKES, Meinolf; HOFFMANN, Ute; MARZ, Lutz: Leitbild und Technik. Zur Entstehung und Steuerung technischer Innovationen. Berlin: Edition Sigma, 1992.

DITLEA, Steve (Hrsg.): Digital Deli – The comprehensive user-lovable menu of computer lore, culture, lifestyles and fancy. New York: Workman Publishing, 1984. Zitiert nach der Online-Ausgabe auf: http://www.atariarchives.org/deli/ [Aufgerufen am 06.11.2004].

EBERT, Christof; RUFFIN, Michel: Produkte entwickeln mit Open-Source-Software – Risiken und Erfahrungen. In: SAUERBURGER, Heinz (Hrsg.): Open-Source-Software. Heidelberg: dpunkt, 2004, S. 27-40.

ETTRICH. Matthias: Koordination und Kommunikation in Open-Source-Projekten. In: GEHRING, Robert A.; LUTTERBECK, Bernd (Hrsg.): Open Source Jahrbuch 2004. Zwischen Softwareentwicklung und Gesellschaftsmodell. Berlin: Lehmanns Media, 2004, S. 79-192.

FISH, Shlomi: The “In for Free Beer” Manifesto. Online veröffentlicht unter: http://fc-solve.berlios.de/oss-fs/in_for_free_beer_manifesto.html [Aufgerufen am 26.01.2005].

FRANKFURTER Allgemeine Zeitung: Linux findet Freunde in Unternehmen. Auch kleinere Anbieter wie Red Hat kommen besser ins Geschäft. 26.06.03, S. 15 (Wirtschaft).

FREIBERGER, Paul; SWAINE, Michael: Fire in the Valley. The Making of the Personal Computer. New York: McGraw-Hill, 1984.

FROMMEL, Oliver: Tagesschau weltweit mit Linux. Das DV-Schnittprogramm Kino bei Tagesschau.de. Linux Magazin, Ausgabe 08/04. München: Linux New Media AG, 08.07.2004. S. 79.

GANCARZ, Mike: Linux and the Unix Philosophy. New York: Elsevier Science, 2003.

GATES III, William Henry: Information at your fingertips 2005. Eröffnungsrede zur Messe „Comdex“ am 14.11.1994. Online archiviert unter: http://www.microsoft.com/billgates/speeches/industry&tech/iayf2005.asp [Aufgerufen am 05.02.2005].

GATES III, William Henry: An Open Letter to Hobbyists. Altair Users’ Newsletter. Albuquerque: 1976. Zitiert nach: http://www.blinkenlights.com/classiccmp/gateswhine.html [Aufgerufen am 09.11.2004].

GEHRING, Robert A.; LUTTERBECK, Bernd (Hrsg.): Open Source Jahrbuch 2004. Zwischen Softwareentwicklung und Gesellschaftsmodell. Berlin: Lehmanns Media, 2004.

GHOSH, Rishab A.; GLOTT, Rüdiger; KRIEGER, Bernhard; ROBLES, Gregorio: Free / Libre and Open Source Software: Survey and Study. Deliverable D 18: Final Report, Part 4: Survey of Developers. Online veröffentlicht unter: http://floss1.infonomics.nl/finalreport/FLOSSfinal-4.pdf. [Aufgerufen am 26.01.2005].

GILLMOR, Dan: We the Media. Grassroots Journalism by the People, for the People. Sebastopol: O’Reilly, 2004.

GRAHAM, Paul: Hackers & Painters. Big Ideas from the Computer Age. Sebastopol: O’Reilly, 2004.

GRASSMUCK, Volker: Freie Software. Zwischen Privat- und Gemeineigentum. Bonn: Bundeszentrale für politische Bildung, 2002.

GRAYLIN, Alvin; KJOLAAS, Kari Anne Hoier; LOFLIN, Jonathan; WALKER, Jimmie Dale: Symbolics, Inc. A failure of heterogeneous engineering. Cambridge, MA: 1998. Online veröffentlicht unter: http://mit.edu/6.933/www/Symbolics.pdf [Aufgerufen am 25.11.2004].

GREVE, Georg C. F.: Geschichte und Philosophie des GNU Projektes. Vortrag an der Universität Paderborn, 05. 12.1998. Online veröffentlicht unter: http://www.gnu.org/philosophy/greve-clown.de.html [Aufgerufen am 09.12.2204].

GRUNWALD, Lukas: Virtuelle Verschlußsache. BSI-VPN SINA wird kommerziell verfügbar. iX Magazin für professionelle Informationstechnik. Ausgabe 09/2004. Hannover: Heise-Verlag, S. 95-97.

HAGEN, Wolfgang: Bill Luhan und Marshall McGates. Die Extension des Menschen als Extension der USA. In: ROESLER, Alexander; STIEGLER, Bernd: Microsoft. Medien, Macht. Monopol. Frankfurt a. M.: Suhrkamp, 2002, S. 11-47.

HANSEN, Ulrich: Hackerlegende gegen Softwarepatente. „Solange wir kämpfen, können wir gewinnen“ – Richard M. Stallman auf Blitzbesuch in Europa. ZDF heute-online vom 27.09.2004. http://www.heute.de/ZDFheute/inhalt/29/0,3672,2197021,00.html [Aufgerufen am 23.11.04].

HÄNTZSCHEL, Jörg: Dirty Harry. Deutsche Potter-Fans basteln an einer Kollektiv-Übersetzung. Süddeutsche Zeitung, 08.07.03. S. 13 (Feuilleton).

HAUBEN, Ronda: Über die Entstehung des Internet und die Rolle der Regierung. In: Telepolis – Magazin der Netzkultur, 24.10.1999. http://www.heise.de/tp/deutsch/inhalt/co/5409/1.html [Aufgerufen am 03.10.2004].

HEINZE, Daniel; KELLER, Alexander: Der Preis der Freiheit – was Softwareentwickler über Open-Source-Lizenzen wissen sollten. In: SAUERBURGER, Heinz (Hrsg.): Open-Source-Software. Heidelberg: dpunkt, 2004, S. 41-48.

HEISE online: Weiter starkes Wachstum bei Anzahl der Websites. 05.08.2004, 10:23 Uhr. http://www.heise.de/newsticker/meldung/49785 [Aufgerufen am 06.08.2004].

HIMANEN, Pekka: Die Hacker-Ethik und der Geist des Informationszeitalters. München: Riemann, 2001.

HOLTGREWE, Ursula: Heterogene Ingenieure: Open Source und Freie Software zwischen technischer und sozialer Innovation. In: GEHRING, Robert A.; LUTTERBECK, Bernd (Hrsg.): Open Source Jahrbuch 2004. Zwischen Softwareentwicklung und Gesellschaftsmodell. Berlin: Lehmanns Media, 2004. S. 339-352.

HORN, Karen: Auf der Allmende des Wissens. Eigentum, Celera Genomics und Microsoft. Frankfurter Allgemeine Zeitung, 14.04.00. S. 13

IANNACCI, Federico: The Linux managing model. In: First Monday, Ausgabe 8, Nr. 12. Dezember 2003. Online veröffentlicht unter: http://firstmonday.org/issues/issue8_12/iannacci/index.html [Aufgerufen am 18.01.2005].

IDC Software Consulting: The Linux Marketplace – Moving From Niche to Mainstream. December 14, 2004. Studie im Auftrag des Open Source Development Labs (OSDL). Online veröffentlicht unter: http://www.osdl.org/docs/linux_market_overview.pdf [Aufgerufen am 16.12.2004].

INTERNET SYSTEMS CONSORTIUM: ISC publishes New Domain Survey Report. Pressemitteilung vom 10.03.2004. http://www.isc.org/about/press/?pr=2004031000 [Aufgerufen am 09.08.2004].

JAKOBS, Joachim: Softwarepatente: Ein schlechter Tag für Europa und für Deutschland. Pressemitteilung der FSFE, 19.05.2004. http://mailman.fsfeurope.org/pipermail/press-release-de/2004q2/000025.html [Aufgerufen am 06.08.2004].

KIESER, Alfred; KUBICEK, Herbert: Organisation. 3. völlig neu bearbeitete Auflage. Berlin; New York: de Gruyter: 1992.

KITTLER, Friedrich: Wissenschaft als Open-Source-Prozeß. Vortrag auf der Konferenz WOS1 (Wizards of OS – Offene Quellen und Freie Software) vom 16.-17.7.1999 im Haus der Kulturen der Welt, Berlin. Im Internet veröffentlicht unter: http://www.hydra.umn.edu/kittler/os.html [Aufgerufen am 18.1.2005].

KNOP, Carsten: Ein Juraprofessor schreckt die Linux-Welt auf. Streit um Rechtsunsicherheiten von Open-Source-Software etwa von Suse oder MySQL. Frankfurter Allgemeine Zeitung, 14.08.03, S. 16 (Wirtschaft).

KREMPL, Stefan: WOS3: Creative Commons als Geheimwaffe der Künstler im Copyright-Krieg. heise-online, 12.06.2004, 11:12 Uhr. http://www.heise.de/newsticker/meldung/48184 Abgerufen am 06.08.2004].

LAKHANI, Karim R.; WOLF, Robert G.: Why Hackers Do What They Do: Understanding Motivation Effort in Free/Open Source Software Projects. In: FELLER, J.;FITZGERALD, B.; HISSAM, S.; LAKHANI, Karim R. (Hrsg.): Perspectives on Free and Open Source Software. Boston: MIT-Press, 2005. S.11. Nach der Online-Ausgabe unter: http://opensource.mit.edu/papers/lakhaniwolf.pdf [Aufgerufen am 26.01.2005].

LANDGERICHT München 1: Urteil vom 19.05.2004 (Aktenzeichen 21 O 6123/03). Zitiert nach der schriftlichen Urteilsbegründung, veröffentlicht durch die Anwaltskanzlei JBB Berlin/München unter: http://www.jbb.de/urteil_lg_muenchen_gpl.pdf [Aufgerufen am 06.08.2004].

LEONARD, Andrew: Open Source Journalism. In salon.com, 8.11.1999, http://www.salon.com/tech/log/1999/10/08/geek_journalism [Aufgerufen am 15.06.2004].

LESSIG, Lawrence: Code und andere Gesetze des Cyberspace. Berlin: Berlin-Verlag, 2001.

LESSIG, Lawrence: The Future of Ideas: The Fate of the Commons in a Connected World. New York: Random House, 2001.

LEVY, Steven: Hackers. Heroes of the Computer Revolution. New York: Penguin, 2001. (erstmals veröffentlicht: 1984)

LINDNER, Roland: Aktie im Blick: IBM. Linux lebt. Frankfurter Allgemeine Zeitung, 13.09.03, S. 21 (Wirtschaft).

LINUX Magazin: Frankreich will neue Open-Source-Lizenz. Ausgabe 09/04. München: Linux New Media AG, 05.08.2004, S. 16.

LINUX Magazin: Open Source oder Free Software. Ausgabe 09/04. München: Linux New Media AG, 05.08.2004. S. 17.

LINUX Magazin: Netfilter-Projekt verteidigt GPL. Ausgabe 06/04. München: Linux New Media AG, 06.05.2004, S. 21.

LINUX Magazin: Red Hat steigert Gewinn. Ausgabe 03/05. München: Linux New Media AG, S. 21.

LISCHKA, Konrad: Als Computer Götter waren. „Spacewar!“ am MIT. In Telepolis – Magazin der Netzkultur, 04.08.2001. http://www.heise.de/tp/deutsch/special/spiel/9223/1.html [Aufgerufen am 0006.09.2004].

LUTHIGER, Benno: Alles aus Spaß. Zur Motivation von Open-Source-Entwicklern. In: GEHRING, Robert A.; LUTTERBECK, Bernd (Hrsg.): Open Source Jahrbuch 2004. Zwischen Softwareentwicklung und Gesellschaftsmodell. Berlin: Lehmanns Media, 2004, S. 93-105.

LUTHIGER, Benno: Fun and Software Development. FASD Study – Provisional Results. Version 1.0. Zürich: Juli 2004. Online veröffentlicht unter: http://www.isu.unizh.ch/fuehrung/blprojects/FASD/FASD_eval1.pdf [Aufgerufen am 26.01.2005].

MARSON, Ingrid: OpenOffice.org goes Swahili. ZDNet UK, December 06, 2004, 16:05 GMT. http://news.zdnet.co.uk/software/linuxunix/0,39020390,39179058,00.htm [Aufgerufen am 11.12.2004].

MAUS, Stephan: Ins Biotop der Hacker-Gemeinde. Chaos, Computer und Club: Eine Existenz zwischen Breitmaulfrosch, Klapperstorch und dem Geist der Maschine. Süddeutsche Zeitung, 12.08.03, S. 13 (Feuilleton).

McKUSICK, Marshall Kirk: Twenty Years of Berkeley Unix. From AT&T-Owned to Freely Redistributable. In: DIBONA, Chris; OCKMAN, Sam; STONE, Mark (Hrsg.): OpenSources. Voices from the Open Source Revolution. Sebastopol: O'Reilly, 1999, S. 31-46.

McMILLAN, Robert: The Great Dictator. Linus Torvalds: The Benevolent, Brilliant Keeper of the Kernel. In: Linux Magazine, San-Francisco: 12/2002. Online veröffentlicht unter: http://www.linux-mag.com/2002-12/linus_01.html [Aufgerufen am 10.01.2005].

McVOY, Larry: A solution for growing pains. E-Mail an die Newsgroup „muc.lists.linux-kernel" vom 30.09.1998 13:27:41 Uhr. Archiviert unter: http://list-archive.xemacs.org/xemacs-beta/199809/msg00589.html [Aufgerufen am 25.01.2005].

MERETZ, Stefan: LINUX & CO. Freie Software – Ideen für eine andere Gesellschaft. Neu-Ulm: AG SPAK, 2000. Online veröffentlicht unter: http://www.kritische-informatik.de/index.htm?fsrevol.htm [Aufgerufen am 13.12.2004].

MEYN, Hermann: Massenmedien in der Bundesrepublik Deutschland. Berlin: Edition Colloquium, 1994

MICROSOFT Corporation: Inside out. Microsoft – In Our Own Words. New York: Warner Books, 2000.

MINK, Andreas: Schmutzige Kosten von Billigpreisen. Zürich: NZZ am Sonntag. 09.11.2003, 02:27, Online kostenpflichtig abrufbar unter: http://nzz.gbi.de/webcgi?START=A20&T_FORMAT=5&DOKM=709962_NZZ_0&WID=49872-8270855-90662_3 [Aufgerufen am 27.01.2005; verwendet wurde eine kostenlose Kopie der Seite aus dem Google-Cache].

MINSKY, Marvin: Television Camera-To-Computer Adapter: PDP-6 Device 770. Artificial Intelligence Project Memo No 75 (AIM-75), January 1965. Online veröffentlicht unter: ftp://publications.ai.mit.edu/ai-publications/pdf/AIM-075.pdf [Aufgerufen am 02.10.2004].

MIRRICK, John: Before Microsoft. 29.09.1996. Webseite der Fakultät für Computerwissenschaft der Polytechnischen Hochschule Virginia: http://ei.cs.vt.edu/~history/Gates.Mirick.html [Aufgerufen am 06.11.2004].

MIT Tech Talk: MIT leaves behind a rich history in Tech Square, 17.03.2004 (Volume 48, Number 21). Zitiert nach der Online-Ausgabe unter: http://web.mit.edu/newsoffice/2004/techsquare-0317.html [Aufgerufen am 02.10.2004].

MÖLLER, Erik: Die heimliche Medienrevolution. Wie Weblogs, Wikis und freie Software die Welt verändern. Hannover: heise, 2004.

MOODY, Glyn: Rebel Code. Linux and the Open Source Revolution. New York: Penguin, 2002.

MOODY, Glyn: The Greatest OS That (N)ever Was. In: Wired Magazine. Ausgabe 5.08, August 1997. Online veröffentlicht unter: http://www.wired.com/wired/5.08/linux_pr.html [Aufgerufen am 18.01.2005].

MOON, Jae Yun; SPROULL, Lee: Essence of distributed work: The case of the Linux kernel. In: First Monday, Ausgabe 5, Nr. 11. November 2000. Online veröffentlicht unter: http://www.firstmonday.org/issues/issue5_11/moon/index.html [Aufgerufen am 21.01.2005].

MORTON, Andrew: Rede zum Thema "The Policy Implications of Open Source Software". Veranstalter: The Forum on Technology & Innovation. 15.07.2004. Online veröffentlicht unter: http://www.tech-forum.org/upcoming/transcripts/Transcript_OpenSource_07-15-04.pdf [Aufgerufen am 18.12.2004].

MURDOCK, Ian: A Brief History of Debian. Appendix A: The Debian Manifesto. 01.06.1994. Online veröffentlicht unter: http://www.debian.org/doc/manuals/project-history/ap-manifesto.en.html [Aufgerufen am 02.02.2005].

NEFF, Berthold: David Linux besiegt Goliath Microsoft. Die Windows-Verkäufer verpokern sich mit Preisnachlässen. Neuer Anbieter hofft auf Signalwirkung. Süddeutsche Zeitung, 30.05.03, S. 40 (München).

NETCRAFT: Slight Linux Market Share Loss for Red Hat. Veröffentlichung des Unternehmens: http://news.netcraft.com/archives/2004/07/12/slight_linux_market_share_loss_for_red_hat.html [Aufgerufen am 25.01.2005].

PERENS, Bruce: The Open Source Definition. In: DIBONA, Chris; OCKMAN, Sam; STONE, Mark (Hrsg.): OpenSources. Voices from the Open Source Revolution. Sebastopol: O'Reilly, 1999, S. 171-188.

PI (pi@helpdesk.rus.uni-stuttgart.de): RMS ueber User Interface Copyrights und Software Patente. In: Chalisti, Onlinemagazin, Ausgabe 15, 24.07.1991. Veröffentlicht unter: http://www.chscene.ch/ccc/chalisti/chalisti_15/008_OFA8.html [Aufgerufen am 08.01.2005].

QUÉAU, Philippe: In search of the common good. The information society and archives. In: CITRA 1998: Access to information. Technological challenges. Proceedings of the thirty-third International Conference of the Round Table on Archives, Paris 1999, S. 17-26. Zitiert nach der Online-Veröffentlichung durch die UNESCO:
http://www.unesco.org/webworld/points_of_views/queau_2.html
[Aufgerufen am 08.08.2004].

RAYMOND, Eric Steven: A Brief History of Hackerdom. In: Ders.: The Cathedral and the Bazaar. Musings on Linux and Open Source by an Accidental Revolutionary. Sebastopol: O'Reilly, 2001, S. 1-18.

RAYMOND, Eric Steven: A Fan of Freedom: Thoughts on the Biography of RMS. 11.11.2003. Online veröffentlicht unter:
http://www.catb.org/~esr/writings/rms-bio.html [Aufgerufen am 13.12.2004].

RAYMOND, Eric Steven: Homesteading the Noosphere. In: Ders.: The Cathedral and the Bazaar. Musings on Linux and Open Source by an Accidental Revolutionary. Sebastopol: O'Reilly, 2001, S. 65-111.

RAYMOND, Eric Steven: How to Become a Hacker. In: Ders.: The Cathedral and the Bazaar. Musings on Linux and Open Source by an Accidental Revolutionary. Sebastopol: O'Reilly, 2001, S. 195-213.

RAYMOND, Eric Steven: Revenge of the Hackers. In: The Cathedral and the Bazaar. Musings on Linux and Open Source by an Accidental Revolutionary. Sebastopol: O'Reilly, 2001, S. 167-192.

RAYMOND, Eric Steven: Rootless Root. The Unix Koans of Master Foo. Online veröffentlicht unter: http://www.catb.org/~esr//writings/unix-koans/ [Aufgerufen am 29.01.2005].

RAYMOND, Eric Steven: Shut Up And Show Them The Code. Veröffentlichung im Online-Magazin Linux Today vom 28.06.1999, 15 :51 Uhr. Online archiviert unter:
http://linuxtoday.com/news_story.php3?ltsn=1999-06-28-023-10-NW-SM
[Aufgerufen am 05.02.2005].

RAYMOND, Eric Steven: Terminology Wars: A Web Content Analysis. 25.07.2004. http://www.catb.org/~esr/writings/terminology/ [Aufgerufen am 08.08.2004].

RAYMOND, Eric Steven: The Art of UNIX Programming. Boston: Addison-Wesley, 2003. Online veröffentlicht unter:
http://www.catb.org/~esr/writings/taoup/html/ch02s01.html
[Aufgerufen am 12.12.2004].

RAYMOND, Eric Steven: The Cathedral and the Bazaar. Musings on Linux and Open Source by an Accidental Revolutionary. Sebastopol: O'Reilly, 2001.

RAYMOND, Eric Steven: The Magic Cauldron. In: The Cathedral and the Bazaar. Musings on Linux and Open Source by an Accidental Revolutionary. Sebastopol: O'Reilly, 2001, S. 134-140.

RAYMOND, Eric Steven: The New Hacker's Dictionary. Boston: MIT-Press, 1996. Zitiert nach der Online-Ausgabe („The Jargon File") unter: http://www.catb.org/~esr/jargon/html/ [Aufgerufen am 31.01.2005].

REITER, Bernhard E.: Wandel der IT. Mehr als 20 Jahre freier Software. In: SAUERBURGER, Heinz (Hrsg.): Open-Source-Software. Heidelberg: dpunkt, 2004, S. 83-110.

RITCHIE, Dennis M.: The Evolution of the Unix Time-sharing System. In: Lecture Notes in Computer Science #79: Language Design and Programming Methodology. Berlin: Springer-Verlag, 1980. Zitiert nach der Online-Veröffentlichung unter: http://cm.bell-labs.com/cm/cs/who/dmr/hist.pdf [Aufgerufen am 12.12.2004].

RIVLIN, Gary: Leader of the Free World. How Linus Torvalds became benevolent dictator of Planet Linux, the biggest collaborative project in history. In: Wired Magazine. Ausgabe 11.11, November 2003, S.157. Online veröffentlicht unter: http://www.wired.com/wired/archive/11.11/linus_pr.html [Aufgerufen am 18.01.2005].

ROBLES, Gregorio; SCHEIDER, Hendrik; TRETKOWSKI, Ingo; WEBER, Niels: Who Is Doing It? A research on Libre Software developers. Berlin: Technische Universität Berlin, Fachgebiet für Informatik und Gesellschaft, August 2001. Online veröffentlicht unter: http://widi.berlios.de/paper/study.html [Aufgerufen am 18.01.2005].

ROESLER, Alexander; STIEGLER, Bernd (Hrsg.): Microsoft. Medien, Macht. Monopol. Frankfurt a. M.: Suhrkamp, 2002.

SAUERBURGER, Heinz (Hrsg.): Open-Source-Software. Heidelberg: dpunkt, 2004.

SCHIRRMACHER, Frank: Die Patente der Fliege. Eine Begegnung mit J. Craig Venter. Frankfurter Allgemeine Zeitung, 18.04.00, S. 49 (Feuilleton).

SCHWÄGERL, Christian: Der Mensch als Markt. Kommerzkalkül: Wenn eine Sekte Babys klont. Frankfurter Allgemeine Zeitung, 28.12.02, S. 29 (Feuilleton).

SCHWÄGERL, Christian: Gute Träume vom manipulierten Leben. Fünfzig Jahre nach der DNS-Entschlüsselung wird das Erbgut zum Gegenstand der Kunst. Frankfurter Allgemeine Zeitung, 28.02.03, S. 44 (Feuilleton).

SHAIKH, Maha; CONFORD, Tony: Version Management Tools: CVS to BK in the Linux Kernel. Online verfügbar unter: http://opensource.mit.edu/papers/shaikhcornford.pdf [Aufgerufen am 25.01.2005].

STALLMAN, Richard Matthew: Copyleft: Pragmatic Idealism. In Ders.: Free Software, Free Society. Selected Essays. Boston: GNU-Press, 2002, S. 91-94.

STALLMAN, Richard Matthew: EMACS: The Extensible, Customizable Display Editor. Online veröffentlicht unter:
http://www.gnu.org/software/emacs/emacs-paper.html
[Aufgerufen am: 22.11.2004].

STALLMAN, Richard Matthew: Free Software Definition. In: Ders.: Free Software, Free Society. Selected Essays. Boston: GNU-Press, 2002, S. 41-44.

STALLMAN, Richard Matthew: Free Software, Free Society. Selected Essays. Boston: GNU-Press, 2002.

STALLMAN, Richard Matthew: Free Software: Freedom and Cooperation. Speech at New York University in New York, New York on 29 May 2001. In: Ders.: Free Software, Free Society. Selected Essays. Boston: GNU-Press, 2002, S. 155-186.

STALLMAN, Richard Matthew: GPL-incompatible license. E-Mail an die Mailingliste forum@XFree86.Org vom 07.02.2004, 04:01:10 Uhr. Online archiviert unter: http://www.xfree86.org/pipermail/forum/2004-February/003974.html
[Aufgerufen am 25.01.2005].

STALLMAN, Richard Matthew: Lecture at Kungliga Tekniska Högskolan (KTH, Königliches Institut für Technologie). Stockholm: 1986. Veröffentlicht online unter: http://www.gnu.org/philosophy/stallman-kth.html
[Aufgerufen am 22.11.2004].

STALLMAN, Richard Matthew; WEINREB, Daniel; MOON, David: Lisp Machine Window Manual. Edition 1.1. System Version 95. Boston: MIT, August 1983. S. 2. Archiviert online unter:
http://www.bitsavers.org/pdf/mit/cadr/LISPM_WindowSystem_Aug83.pdf
[Aufgerufen am 10.10.2005].

STALLMAN, Richard Matthew: LISP Machine ZMail Manual. First Edition, ZMail Version 5.0 (System 94). Boston: MIT, März 1983. S. 1. Archiviert online unter: http://www.bitsavers.org/pdf/mit/cadr/LISPM_ZmailMan_Apr83.pdf
[Aufgerufen am 10.10.2005].

STALLMAN, Richard Matthew: My Lisp Experiences and the Development of GNU Emacs. Mitschrift einer Rede Stallmans auf der „International Lisp Conference“ am 28.10.2002. http://www.gnu.org/gnu/rms-lisp.html
[Aufgerufen am 02.12.2004].

STALLMAN, Richard Matthew: New UNIX implementation. Mail from Tue, 27-Sep-83 12:35:59 EST to net.unix-wizards,net.usoft. Veröffentlicht unter:
http://www.gnu.org/gnu/initial-announcement.html
[Aufgerufen am 04.12.2004].

STALLMAN, Richard Matthew: RMS responds. E-Mail an das Webforum des Online-Magazins Slashdot vom 28.06.1999, 08:03 Uhr. Online archiviert unter: http://slashdot.org/article.pl?sid=99/06/28/1311232&mode=thread [Aufgerufen am 02.02.2005].

STALLMAN, Richard Matthew: The GNU Manifesto. In: Ders.: Free Software, Free Society. Selected Essays. Boston: GNU-Press, 2002, S. 31-39.

STALLMAN, Richard Matthew: The GNU Operating System and the Free Software Movement. In: DIBONA, Chris; OCKMAN, Sam; STONE, Mark (Hrsg.): OpenSources. Voices from the Open Source Revolution. Sebastopol: O'Reilly, 1999, S. 53-70.

STALLMAN, Richard Matthew: The GNU Project. In Ders.: Free Software, Free Society. Selected Essays. Boston: GNU-Press, 2002, S. 15-30.

STALLMAN, Richard Matthew: Why "Free Software" is Better than "Open Source". 1998. In Ders.: Free Software, Free Society. Selected Essays. Boston: GNU-Press, 2002, S. 55-60.

STALLMAN, Richard Matthew: Why "Free Software" is better than "Open Source". In: DIBONA, Chris; OCKMAN, Sam; STONE, Mark (Hrsg.): OpenSources. Voices from the Open Source Revolution. Sebastopol: O'Reilly, 1999, S. 55-60.

STALLMAN, Richard Matthew: Why Software Should Be Free. In: Ders.: Free Software, Free Society. Selected Essays. Boston: GNU-Press, 2002, S. 119-132.

STERLING, Bruce: The Hacker Crackdown. Law and Disorder On The Electronic Frontier. New York: Bantam Books, 1993.

STOLL, Clifford: Das Kuckucksei. Die Jagd auf die deutschen Hacker, die das Pentagon knackten. Frankfurt am Main: Fischer, 1990.

TANENBAUM, Andrew Stephen: Frequently Asked Questions. Veröffentlichung auf seiner Homepage unter: http://www.cs.vu.nl/~ast/home/faq.html [Aufgerufen am 19.01.2005].

TANENBAUM, Andrew Stephen: Re: Unhappy campers. Mail an die Newsgroup comp.os.minix vom 5 Feb 92 23:23:26 GMT. Online veröffentlicht unter: http://www.chrisshort.net/archives/open-source-software/linux-is-obsolete/ [Aufgerufen am 10.01.2005].

TANENBAUM, Andrew Stephen: Unhappy campers. Mail an die Newsgroup comp.os.minix vom 3 Feb 92 22:46:40 GMT. Online veröffentlicht unter: http://www.chrisshort.net/archives/open-source-software/linux-is-obsolete/ [Aufgerufen am 10.01.2005].

TANENBAUM, Andrew Stuart: Operating Systems: Design and Implementation. Englewood Cliffs, N.J.: Prentice-Hall, 1987.

The Tanenbaum-Torvalds Debate. In: DIBONA, Chris; OCKMAN, Sam; STONE, Mark (Hrsg.): OpenSources. Voices from the Open Source Revolution. Sebastopol: O'Reilly, 1999, S. 223 (Appendix A).

TORVALDS, Linus Benedict: Free minix-like kernel sources for 386-AT. Mail an die Newsgroup comp.os.minix vom 05.10.1991 09:24:25 PST. Archiviert unter anderem bei:
http://www.chrisshort.net/archives/open-source-software/reference/05-Oct-1991-free-minix-like-kernel-sources-for-386-at.html
[Aufgerufen am 10.01.2005].

TORVALDS, Linus Benedict: Just for fun. Wie ein Freak die Computerwelt revolutionierte. München/Wien: Hanser, 2001.

TORVALDS, Linus Benedict: Linux History. Zusammenfassung von Ankündigungen und den Linux-Kernel begleitenden Notizen, verfaßt am 31.07.1992. Veröffentlicht auf: http://www.li.org/linuxhistory.php [Aufgerufen am 08.12.2004].

TORVALDS, Linus Benedict: Re: 2.1.123 and fbcon.c. E-Mail an die Newsgroup „muc.lists.linux-kernel" vom 29.09.1998, 23:11:48 Uhr. Online archiviert unter: http://www.ussg.iu.edu/hypermail/linux/kernel/9809.3/0850.html
[Aufgerufen am 25.01.2005].

TORVALDS, Linus Benedict: Re: Coding style – a non-issue. Mail an die Newsgroup „mlist.linux.kernel" vom 30.11.2001, 22:00:26 PST. Online archiviert unter:
http://groups.google.de/groups?hl=de&lr=&selm=linux.kernel.Pine.LNX.4.33.0111302048200.1459-100000%40penguin.transmeta.com
[Aufgerufen am 25.01.2005].

TORVALDS, Linus Benedict: Re: Shared Interrupts – PCI. E-Mail an die Newsgroup „linux.kernel" vom 29.06.1995 21:03:03 Uhr. Online archiviert unter:
http://www.uwsg.iu.edu/hypermail/linux/kernel/9506/0194.html
[Aufgerufen am 24.01.2005].

TORVALDS, Linus Benedict: Re: Unhappy campers. Mail an die Newsgroup comp.os.minix vom 6 Feb 92 10:33:31 GMT. Online veröffentlicht unter:
http://www.chrisshort.net/archives/open-source-software/linux-is-obsolete/
[Aufgerufen am 10.01.2005]. [Re: Unhappy campers, a.a.O.]

TORVALDS, Linus Benedict: Release notes for Linux v0.95. Veröffentlicht am 07.03.1992. Online abrufbar unter anderem unter:
http://source.rfc822.org/linux/kernel/Historic/old-versions/RELNOTES-0.95
[Aufgerufen am 19.01.2005].

TORVALDS, Linus Benedict: The Linux Edge. In: DIBONA, Chris; OCKMAN, Sam; STONE, Mark (Hrsg.): OpenSources. Voices from the Open Source Revolution. Sebastopol: O'Reilly, 1999, S. 101-111.

TORVALDS, Linus Benedict: What would you like to see most in minix? Mail an die Newsgroup comp.os.minix vom 25.08.1992 23:12:08 Uhr PST. Online archiviert unter: http://groups.google.de/groups?hl=de&lr=&selm=1991Aug25.205708.9541%40klaava.Helsinki.FI [Aufgerufen am 06.01.2005].

UYTTERHOEVEN, Geert. The Linux Frame Buffer Device Penguin Logo. Online veröffentlicht unter: http://members.chello.be/cr26864/Linux/fbdev/logo.html [Aufgerufen am 26.01.2005].

WALLACE, James: Hard Drive: Bill Gates and the Making of the Microsoft Empire, New York: HarperCollins Publishers, 1993.

WHEELER, David A.: Why Open Source Software / Free Software (OSS/FS, FLOSS, or FOSS)? Look at the Numbers! Online veröffentlicht unter: http://www.dwheeler.com/oss_fs_why.html [Aufgerufen am 25.01.2005].

WILLIAMS, Sam: Free as in freedom. Richard Stallman's Crusade for Free Software. Sebastopol: O'Reilly, 2002.

WIRZENIUS, Lars: Linux Anecdotes. Rede für die Linux Expo. 27.04.1998. Online veröffentlicht unter: http://liw.iki.fi/liw/texts/linux-anecdotes.html [Aufgerufen am 18.12.2004].

WOZNIAK, Stephen: Homebrew and how the Apple came to be. In: DITLEA, Steve (Hrsg.): Digital Deli – The comprehensive user-lovable menu of computer lore, culture, lifestyles and fancy. New York: Workman Publishing, 1984. Zitiert nach der Online-Ausgabe auf: http://www.atariarchives.org/deli/ [Aufgerufen am 06.11.2004].

YAMAGATA, Hiroo: The Pragmatist of Free Software. Linus Torvalds Interview. In: Wired Japan, 20.09.1997. Online veröffentlicht unter: http://kde.sw.com.sg/food/linus.html [Aufgerufen am 21.01.2005].

ZIMMERMANN, Thomas. Open Source und Freie Software – Soziale Bewegung im virtuellen Raum. In: GEHRING, Robert A.; LUTTERBECK, Bernd (Hrsg.): Open Source Jahrbuch 2004. Zwischen Softwareentwicklung und Gesellschaftsmodell. Berlin: Lehmanns Media, 2004. S. 353-368.

Anhang

Das Schwein guckt ins Uhrwerk oder: Was ist Quellcode?

Die Prozessoren von Computern erhalten und verarbeiten ihre Instruktionen in Form von Nullen und Einsen, dem Binärcode, der auch als *Maschinensprache* bezeichnet wird.[678] Die Maschinensprache ist vom Prozessor abhängig, d.h. ein PowerPC-Prozessor (eines Mac) versteht nicht die Maschinensprache eines i386-Prozessors (eines PC).

Da kein Mensch lange Ketten von Nullen und Einsen schreiben (bzw. per Hand Lochkarten lochen) will, wurde schon in den 50ern eine für Menschen schreib- und lesbare Form der Maschinensprache erfunden, die *Assemblersprache*, die es ermöglicht, statt „10110000 01100001“ den Befehl „mov al, 0x61“ zu schreiben.[679] Instruktionen in Assemblersprache sind simpel, was nicht heißt, daß sie einfach zu verstehen sind, sondern bedeutet, daß sie keine komplexeren Aufgaben erfüllen können, als z.B. Werte von einem Speicherort im Prozessor zu einem anderen zu verschieben, sie zu addieren oder zu vergleichen. Die Assemblersprache ist ebenso prozessorabhängig wie die Maschinensprache.[680] Zum Überführen ihrer Instruktionen in die Maschinensprache dient ein spezielles Übersetzungsprogramm, der „Assembler“.

Die Ausgabe von Text auf einem Bildschirm in Assemblersprache zu programmieren, wäre ebenso mühsam wie eine Autofahrt von Mainz nach Dortmund im ersten Gang. Aus diesem Grund gibt es *Programmiersprachen*, wie zum Beispiel C, Lisp, Java, Perl oder Basic, die für Standardaufgaben wie die Textausgabe bereits Befehle („printf“ in C) mitbringen. Programmiersprachen bieten vorgefertigte Wege für Problemlösungen, sind – nach einer menschlichen Logik – besser strukturierbar und in unterschiedlichem Maße auch für Nicht-Experten verständlich.

Hat man ein Programm in einer Programmiersprache fertig geschrieben, muß es, damit es der Computer ausführen kann, in Maschinensprache übersetzt werden. Dafür gibt es zwei Wege: Entweder die Programme werden von einer Art Dolmetscher-Software, dem „*Interpreter*“ gleichzeitig Wort für Wort übersetzt, während sie ausgeführt werden („zur Laufzeit“).[681] Oder sie werden zunächst im Ganzen und „in einem Rutsch“ von einem *Compiler* übersetzt (kompiliert): Resultat des Kompilierens ist eine Datei in Maschinensprache, die separat zu einem beliebigen Zeitpunkt auf dem Computer ausgeführt werden

---

[678] Maschinensprache wird in Dateien in Form von hexadezimalen Werten dargestellt, also etwa „AFFE0815“ für „1010.1111.1111.1110.0000.1000.0001.0101“. Man sollte sich also nicht darüber wundern, daß man in seinem Computer noch nie Dateien voller Nullen und Einsen gefunden hat.

[679] Vgl. Wikipedia: Assembly Language. In: http://en.wikipedia.org/wiki/Assembly_language [Aufgerufen am 11.12.20004]. Bei dem in Maschinensprache und in Assemblersprache zitierten Befehl handelt es sich um die Anweisung, den hexadezimalen Wert 61 (im Dezimalsystem 91) in ein Register namens „al“ des Prozessors zu schreiben.

[680] Lange Zeit wurden Betriebssysteme, viele Programme und die meisten Videospiele in Assemblersprache geschrieben; die hardwarenahe Sprache hat den Vorteil, die Leistungsfähigkeit des Prozessors gut auszunutzen. Noch heute sind deshalb viele Teile von Programmen sowie „Treiber“ etwa zur Ansteuerung eines Druckers in Assemblersprache verfaßt. Der Vorteil der Sprache ist auch ihr Nachteil: Sie gilt nur für einen Prozessor, in ihr verfaßte Programme sind nicht übertragbar ("portabel“).

[681] Beispiele dafür sind Javascript, Perl oder Basic. In gewissem Sinne kann man auch HTML als interpretierte Sprache verstehen – dabei dient der Webbrowser als Interpreter.

kann.[682] Komplexe Programme wie Betriebssysteme oder Textverarbeitungen werden normalerweise kompiliert.[683]

Die unten stehende Grafik zeigt den Weg vom Schreiben eines Programms über die Verarbeitungsschritte des Compilers bis hin zur ausführbaren Datei.

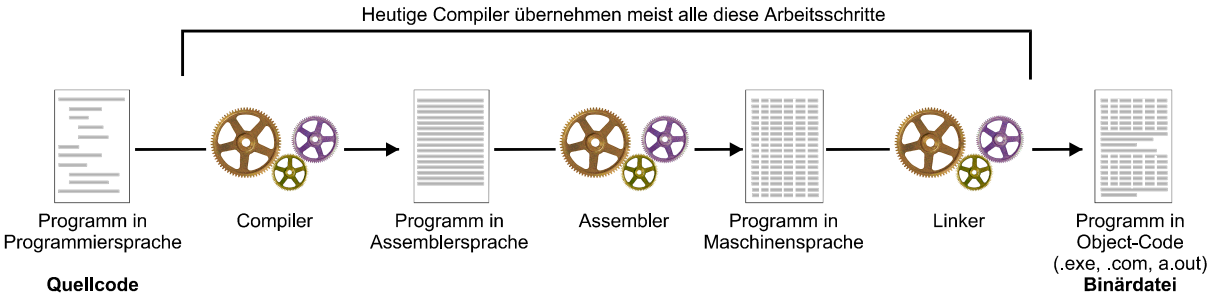


Abbildung 31: Vom Quellcode zur Binärdatei – der Weg durch den Compiler

Ein Programm, das in der lesbaren Form einer Programmiersprache vorliegt, wird auch als *Quellcode* (engl. „Source“) bezeichnet. Das gleiche Programm in Maschinensprache wird als *Binärdatei* bezeichnet.

Ist die Binärdatei einmal kompiliert, so ist es fast unmöglich, daraus die Quelldatei zu rekonstruieren. Zwar gibt es entsprechende Instrumente, wie Disassembler, der Weg zurück ist jedoch sehr langwierig und oft ohne brauchbare Ergebnisse. Das Verhältnis zwischen Quellcode und Binärdatei wird deswegen auch oft mit einem Rezept und einem Kuchen verglichen: Das Erkennen der Zutaten nur durch Probieren des fertigen Gebäcks mag Experten bei einem Streuselkuchen noch möglich sein; bei der Hochzeitstorte wäre es dagegen – auch aus sozialen Gründen – sinnvoller, die Schwiegermutter nach dem Rezept zu fragen.

Softwarehersteller, die ihre Programme als Ware betrachten, verkaufen meist nur Datenträger mit den Binärdateien und behalten den Quellcode als Betriebsgeheimnis für sich. Solche Software wird auch als *„proprietär“*[684] oder als „geschlossener Code“ bezeichnet. Bei freier und Open-Source-Software muß dagegen der Quellcode grundsätzlich mitgeliefert oder auf Anfrage zur Verfügung gestellt werden. Solche Software kann jederzeit geändert oder erweitert werden, indem man den Quellcode umschreibt und ihn anschließend neu kompiliert, bzw. das von einem Experten tun läßt.[685] Bei proprietärer Software ist man dagegen darauf angewiesen, daß der Hersteller die gewünschten Änderungen vornimmt, was oft weder schnell und günstig, noch sehr wahrscheinlich ist.

---

[682] Genau genommen besteht die endgültige Datei aus „Objekt Code“, d.h. die Datei durchläuft einen weiteren Verarbeitungsschritt der als „Linker“ oder „Loader“ bezeichnet wird; dieser fügt zu den Anweisungen in Maschinensprache weitere Informationen hinzu – etwa über Verknüpfungen zu anderen Dateien (libraries) oder über den Speicher-Adressraum, in dem das Programm ausgeführt werden soll. Dateien in Object Code haben oft Endungen wie “.exe“ und “.com“ (DOS) oder „a.out“ (Linux); obwohl sie nicht nur aus Nullen und Einsen bzw. hexadezimalen Werten bestehen, werden sie „Binaries“, also Binärdateien genannt.

[683] Den Unterschied zwischen Compilern und Interpretern verwischen sogenannte „Just-in-time-Compiler“ – sie kompilieren das ganze Programm in Sekundenbruchteilen, bevor sie es ausführen und erwecken so den Eindruck von Interpretern. Prominentes Beispiel dafür ist die Programmiersprache Perl.

[684] Nach dem lateinischen „proprietas“, d.h. „Eigentum“.

[685] Viele Anwender von GNU/Linux haben dies schon getan, wenn sie einen angepaßten Kernel erzeugt haben.

Interview mit Joseph Weizenbaum

Abbildung 32: Joseph Weizenbaum am 11. Februar 2005 auf dem Balkon seiner Wohnung in Berlin.
Foto: Ulrich Hansen

Frage: „Sie haben in Ihrem Buch „Computer Power and Human Reason“ auch über die Hacker am MIT geschrieben. Als ‚computer bums’, also zwanghafte Programmierer haben Sie sie damals beschrieben und das ist, glaube ich, schlecht aufgenommen worden.“

Weizenbaum: „Ja.“

Frage: „Ich habe das Buch „Hackers“ von Steven Levy gelesen...“

Weizenbaum: (lächelt) „Ich auch.“

Frage: „...und er schreibt, Richard Greenblatt, einer der damaligen Hacker, hätte sich bei Ihnen persönlich beschwert und Ihr Kollege Marvin Minsky habe Sie ziemlich harsch kritisiert: Sie hätten die Hacker-Community damals nicht verstanden.“

Weizenbaum: „Vielleicht habe ich das Wort ‚Hacker’ damals benutzt, ich erinnere mich nicht. Aber ein entscheidendes Charakteristikum von ‚computer bum’, wie ich das genannt habe, ist, daß sie nie etwas fertig machen. Die kennen sich aus mit dem Computer, die können sehr viel. Aber wenn zum Beispiel eine Bank sie anstellt und sie eine Aufgabe bekommen – sie sollen zum Beispiel ein Programm schreiben – dann geschieht folgendes: Erstens werden sie ein Programm schreiben, wenn überhaupt, das niemand anderes überhaupt verstehen kann oder bedienen kann oder korrigieren kann oder lesen kann. Aber zweitens werden sie damit auch nicht fertig. Sie werden sagen: ‚Ah jetzt habe ich dieses gemacht, dann kann ich auch noch jenes machen.’ Und bald sind sie dabei, ein System herzustellen, das sich nicht nur um die Finanzen kümmert, sondern auch noch um wer weiß was anderes.

Also: Leute wie Greenblatt, der eine schöne LISP-Maschine gebaut hat, die nenne ich nicht Computer bums – überhaupt nicht. Überhaupt nicht! Es ist interessant, wie viele von den Leuten da in dem KI Lab im MIT zu mir gekommen sind und gesagt haben ‚Du meinst mich’ oder ‚Du hast mich als Modell genommen’. Die haben sich selbst damit i-

dentifiziert! Aber die meinte ich ja gar nicht, die tatsächlich etwas machen. Ein richtiger ‚computer bum' als graduate student im AI Lab im MIT würde da nicht ein Jahr lang bleiben können. Er würde nämlich nichts schaffen. Er würde einfach nichts schaffen, er würde nie etwas zu Ende bringen. Er würde vielleicht wunderbare, clevere Programme schreiben, aber sie hätten keinen Nutzen."

Frage: „Haben Sie das damals unterschätzt, daß sich damals auch eine Art soziale Community gebildet hat?"

Weizenbaum: „Ich glaube, im Bereich des Computers habe ich alles unterschätzt. Auf der einen Seite die rasante Entwicklung, in einem gewissen Sinn aber auch die Slugishness, also die Zähigkeit der Mensch-Maschine Entwicklung – die habe ich auch unterschätzt. Wir haben so viele Fortschritte gemacht, aber das Mensch-Maschine Interface hat sich kaum geändert, in den letzten dreißig Jahren. Der Mensch sitzt da, wie vor dreißig Jahren. Das, zum Beispiel, habe ich auch unterschätzt. Also: Ja. Ich habe es unterschätzt."

Frage: „Was halten Sie denn insgesamt von diesen Projekten? Wenn man sich ein bißchen damit beschäftigt, ist es ja sehr faszinierend, wie offen da gesprochen wird, wie alles miteinander ausgetauscht wird, bei diesen ‚Open Source'-Programmen. Da geht es ja auch um Beherrschbarkeit: Jeder darf etwas ändern, jeder darf etwas frei verbreiten. Das sind ja auch gesellschaftliche Werte, die da eine Rolle spielen: Der Jura-Professor Lawrence Lessig sagt etwa, nur dadurch, daß etwas offen als Code vorliegt, ist es auch gesellschaftlich zu kontrollieren."

Weizenbaum: „Wenn überhaupt!"

Frage: „Wenn überhaupt, ja natürlich. Was halten Sie von dieser Entwicklung, dieser Parallelentwicklung, die da stattgefunden hat?"

Weizenbaum: „Naja, ich halte es für sehr positiv und ganz besonders in einem sozialen, man könnte auch sagen: politischen und kulturellen Sinn. Ich glaube, daß die Praxis der Autonomie in jedem menschlichen Bereich sehr wichtig ist und ein Vorbild sein sollte. Also das ist alles sehr positiv. Aber wenn es darum geht, Computersysteme durchschaubar zu machen, dann ist es kaum eine Hilfe, denke ich. Es ist nämlich so, daß die meisten großen Systeme, die tatsächlich die Arbeit der Welt tun, völlig undurchschaubar sind. Niemand versteht sie im Ganzen. Niemand. Es ist hoffnungslos, sie sind ‚beyond understanding'. Das ist eine gefährliche Situation. Jetzt kann zwar jeder eingreifen und etwas ändern – zuerst nur für sich selbst, später im Tausch mit jemand anderem – aber daß der andere sich dann die Mühe macht, die Änderungen, die da eingefügt werden, zu verstehen, das ist sehr schwer zu glauben. Da ist die Durchschaubarkeit von Systemen eher weiter gefährdet."

Bei dem hier wiedergegebenen Text handelt es sich um einen Ausschnitt aus einem 45-minütigem Fernsehinterview, das ich im Auftrag des ZDF mit Prof. Dr. Joseph Weizenbaum am 11. Februar 2005 in Berlin geführt habe.

Das komplette Interview ist als Video abrufbar unter:
mms://wstreaming.zdf.de/zdf/zdf/050308_weizenbaum_h.wmv
[Aufgerufen am 15.10.2005].

Nachweis der Diagramme

Diagramm 1: Die Zahl der Codezeilen im Linux-Kernel.

Quelldaten für das Diagramm:

Jahr / Version	Codezeilen	Quelle
1991 (0.01)	10.239	Wikipedia. Timeline of Linux development. http://en.wikipedia.org/wiki/Timeline_of_Linux_development [Aufgerufen am 18.01.2005]
1992 (0.96)	40.000	McHUGH, Josh: For the love of hacking. Infobox: Linux: the making of a global hack. Forbes online, 08.10.1998. http://www.forbes.com/forbes/1998/0810/6203094s1.html [Aufgerufen am 18.01.2005].
1993 (0.99)	100.000	McHUGH, a.a.O.
1994 (1.0.0)	176.250	Wikipedia, a.a.O.
1995 (1.2.0)	310.950	Wikipedia, a.a.O.
1996 (2.0.0)	777.956	Wikipedia, a.a.O.
1997 (2.1.0]	800.000	McHUGH, a.a.O.
1998 (2.1.110)	1.500.000	McHUGH, a.a.O.
1999 (2.2.0)	1.800.847	Wikipedia, a.a.O.
2000 (2.2.15)	2.300.000	Geschätzt (nach dem Diagramm in: DIEDRICH, Oliver: Happy Birthday, Tux! Zehn Jahre Freies Betriebssystem. In: c't Magazin für Computertechnik. Ausgabe 19/2001. Hannover: heise, 10.09.2001. S. 166. Online abrufbar unter: http://www.heise.de/ct/01/19/162/default.shtml [Aufgerufen am 18.01.2005].
2001 (2.4.0)	3.377.902	Wikipedia, a.a.O.
2002 (2.4.19)	4.078.662	Nach: The linux-kernel mailing list FAQ. Section 15 – Programming Religion. Online veröffentlicht unter: http://lkml.org/faq/lkmlfaq-15.html [Aufgerufen am 19.01.2005].
2003 (2.6.0)	5.929.913	Wikipedia, a.a.O.

Diagramm 2: Die Zahl der Programmierer des Linux-Kernels.

Quelldaten für das Diagramm:

Jahr / Version	Anzahl	Quelle
1991 (0.0.1)	1	Linus Torvalds alleine.
1992 (0.96a)		keine Angabe
1993 (0.99.11)		keine Angabe
1994 (1.0)	80	DIEDRICH, Oliver: Happy Birthday, Tux! Zehn Jahre Freies Betriebssystem. In: c't Magazin für Computertechnik. Ausgabe 19/2001. Hannover: heise, 10.09.2001. S. 168. Online abrufbar unter: http://www.heise.de/ct/01/19/162/default.shtml [Aufgerufen am 18.01.2005].
1995 (1.2)	128	DIEDRICH, a.a.O.
1996 (2.0)	190	DIEDRICH, a.a.O.
1997 (2.1.0)	196	Eigenrecherche in den Kernelquellen auf: ftp://ftp.kernel.org/pub/linux [Aufgerufen am 18.01.2005].
1998 (2.1.110)	246	Eigenrecherche
1999 (2.2)	269	DIEDRICH, a.a.O.
2000 (2.2.15)	287	Eigenrecherche
2001 (2.4)	375	DIEDRICH, a.a.O.
2002 (2.4.19)	416	Eigenrecherche
2003 (2.6.0)	447	Eigenrecherche

GNU GENERAL PUBLIC LICENSE[686]

Version 2, June 1991
Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place – Suite 330, Boston, MA 02111-1307, USA
Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

---

[686] Dieser Anhang dient zur Information über die GPL. Die vorliegende Arbeit steht nicht unter dieser Lizenz. Die GPL wird zitiert nach: http://www.gnu.org/licenses/licenses.html#GPL [Aufgerufen am 10.01.2005].

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The “Program”, below, refers to any such program or work, and a “work based on the Program” means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term “modification”.) Each licensee is addressed as “you”.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program’s source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the

distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License.

Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

One line to give the program's name and an idea of what it does.

```
Copyright (C) yyyy name of author

This program is free software; you can redistribute it and/or modify it
under the terms of the GNU General Public License as published by the
```

```
Free Software Foundation; either version 2 of the License, or (at your
option) any later version.

This program is distributed in the hope that it will be useful, but
WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General
Public License for more details.

You should have received a copy of the GNU General Public License along
with this program; if not, write to the Free Software Foundation, Inc.,
59 Temple Place – Suite 330, Boston, MA 02111-1307, USA.
```

Also add information on how to contact you by electronic and paper mail. If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) year name of author

Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type ‘show w’.
This is free software, and you are welcome to redistribute it under cer-
tain conditions; type ‘show c’ for details.
```

The hypothetical commands ‘show w’ and ‘show c’ should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than ‘show w’ and ‘show c’; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a “copyright disclaimer” for the program, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright interest in the program
‘Gnomovision’ (which makes passes at compilers) written by James Hacker.

signature of Ty Coon, 1 April 1989

Ty Coon, President of Vice
```

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.