

# Geschichten aus grauer Vorzeit (Teil 1)

[Rüdiger Berlich](#)

*Viele richten sich ihr Linux ein und machen sich weiter keine Gedanken darüber. Für alle, die ein wenig mehr über seine Geschichte erfahren wollen, stellt Rüdiger Berlich die wichtigsten Ereignisse und Motive vor, die die Entwicklung dieses freien Betriebssystems geprägt haben.*

Über die letzten Jahre hinweg hat Linux mächtigen Eindruck bei der Industrie hinterlassen. Trotzdem, seine Geschichte reicht schon fast ein Jahrzehnt - und wenn man die Aktivitäten der Free Software Foundation dazuzählt - sogar mehr als 17 Jahre zurück. Linux fußt, wenn schon nicht auf dem Quellcode, dann doch auf den Paradigmen der Unix-Systeme. Viele sehen in Linux heutzutage eher eine exotische, einigermaßen suspekte Neuheit als eine erwiesenermaßen solide Unix-artige Lösung. Linux an sich ist nicht so besonders neuartig, aber das Ausmaß seiner Popularität ist es!

Um die Wichtigkeit von Linux für die Computerindustrie richtig abschätzen zu können, muß man sich wahrscheinlich in das Jahr 1969 zurückversetzen. Damals begann Ken Thompson bei den Bell Laboratories, einer Tochterfirma von AT&T und Western Electric, mit den Arbeiten am ersten Unix-System überhaupt, um die Beschränktheit der damals populären Batchsysteme zu überwinden. Diese allererste Version wurde in DEC PDP-7 Assemblersprache geschrieben. Um eine maschinenunabhängigere Architektur für sein Betriebssystem zu bekommen, entwickelte Thompson die Programmiersprache B. Dennis Ritchie machte daraus dann später C. Die zweite Version von Unix, diesmal für eine PDP-11, erschien 1971 und war weitgehend in C geschrieben. Die meisten der aktuellen Betriebssysteme werden heutzutage in dieser Sprache geschrieben, allerdings normalerweise in einem etwas moderneren "Dialekt", ANSI C.

Ein Unix-System setzt sich neben dem Betriebssystemkernel auch aus einer ganzen Reihe von Anwendungsprogrammen zusammen. Der Kernel ist für die Kommunikation mit der Hardware einer Maschine und für solche Aufgaben wie die Prozeß- und Speicherverwaltung und vieles andere mehr zuständig. Die Anwendungsprogramme erlauben dem Benutzer das Browsen durch das Dateisystem, das Löschen oder Anlegen neuer Dateien und das Arbeiten mit Textdateien. Solche Anwendungen sind meist klein und sehr spezialisiert, so daß sie mit so wenig Code wie möglich auskommen. Daraus resultiert eine geringe Fehlerhäufigkeit im Code selbst. Die wiederum macht einen der Hauptgründe für die bemerkenswerte Stabilität und einfache Handhabung eines Unix-Systems aus. Anwendungsprogramme können in einer Art von Kette hintereinander aufgereiht werden, wobei die Ausgabe des einen Programms die Eingabe des nächsten bildet. So können auch die schwierigsten Aufgaben gelöst werden, indem solche einfach zu handhabenden Anwendungen geschickt verwendet werden.

Fast alle heutigen kommerziellen Unix-Systeme sind Abkömmlinge von Unix I Version 7, das 1979 erschien. Um diese Zeit herum gabelte sich die Entwicklung von Unix in mehrere Äste. Die beiden wichtigsten sind System V (oder auch Unix V) und BSD. BSD wurde an der University of Berkeley in Kalifornien entwickelt. Für manche, die zu diesen Zeiten vielleicht noch gar nicht auf der Welt waren, ziemlich überraschend, gab es sogar einen Unix-Sproß (XENIX), dessen Entwicklung von Microsoft angestoßen und unterhalten wurde. Allerdings ist Microsoft mittlerweile nicht mehr direkt daran beteiligt. Heutzutage gibt es viele kommerzielle Unixe. Die wohl bekanntesten sind IBM AIX, SunOS, SunSolaris, CompaqTrue64, HP-UX und SGI Irix. Ferner gibt es noch einige BSD Varianten, die frei erhältlich sind.

Ein anderes Unix-artiges Betriebssystem wird im Laufe dieser Geschichte noch eine Rolle spielen, Minix. Minix wurde im Januar 1987 von Andrew S. Tanenbaum vorgestellt und sollte eigentlich als ein eine Art Lehrsystem dienen. Andrew Tanenbaum ist ein berühmter Informatiker, der jetzt an der Vrije Universiteit in Amsterdam arbeitet.

# Standardisierung und der Betriebssystemmarkt

Schaut man sich die Vielfalt der verschiedenen Unix-Implementierungen an, ist es nicht weiter verwunderlich, daß Bedarf nach Standardisierung besteht. Verschiedene Systemhäuser setzten verschiedene Schwerpunkte und waren oft auch gezwungen, neue, manchmal inkompatible, Features zu integrieren, um sich vom Rest des Marktes abzugrenzen. Im Lauf der Zeit wurden im wesentlichen drei Initiativen im Hinblick auf Standardisierung ergriffen. Dies waren SVID (System V Interface Definition), POSIX (früher "/usr/group") und X/Open (früher "BISON"). X/Open ist der jüngste Versuch, auf Quellcodebasis eine Kompatibilität zwischen den beteiligten Unix-Implementierungen herzustellen. Verwunderlicherweise begann diese Initiative in Europa, obwohl sich die US Firmen (einschließlich AT&T) später dazugesellten. Leider bleiben trotz all dieser Anstrengungen immer noch viele Unterschiede zwischen den einzelnen Implementierungen.

In den späten Achtzigern und frühen Neunzigern drangen Intel-basierte Computer und das Betriebssystem Windows in allen seinen vielfältigen Erscheinungsformen in den Markt ein. Sie eroberten den größten Teil des Desktopmarktes und einen wichtigen Anteil am Servermarkt für sich. Somit machten sie Unix in seinem angestammten Einsatzgebiet Konkurrenz. Mittlerweile können Intel und Intel-kompatible Maschinen für sich in Anspruch nehmen, in der gleichen Liga zu spielen wie spezielle Unix Workstations ähnlicher Rechenkapazität. Immer noch sind die meisten der kommerziellen Unixe nicht für Intel Hardware verfügbar. Viele Unix Hersteller kooperieren eng mit den Herstellerfirmen der Chips, die sie verwenden, wenn sie diese nicht gar besitzen. Folglich haben sie ein persönliches Interesse daran, ihre Hardware zu fördern. SGI und die MIPS Prozessorfamilie ist ein Beispiel. Es gibt nur wenige kommerzielle Unix Implementierungen für Intel Maschinen.

An dieser Stelle wird Linux zu einer wichtigen Größe für die Geschäftswelt. Nicht nur, daß Linux ein Unix-kompatibles Betriebssystem ist, das Microsoft Windows auf seinen angestammten Intel Plattformen herausfordert. Es läuft auch auf allen wichtigen Hardware Plattformen, angefangen bei Handhelds mit Motorola 68000 Prozessoren über alle Intel (-kompatiblen) PCs bis hin zu den IBM S/390 Mainframes. Weil dieselbe API (Application Programming Interface) auf allen Plattformen verwendet wurde, besteht zwischen allen Plattformen völlige Kompatibilität auf Quellcodeebene. Einmal gelernt - überall angewandt ... . Linux erreicht das, was SVID, POSIX und X/Open für Unix zu erreichen versucht haben, nämlich eine gemeinsame Codebasis für alle Plattformen. Für einen altgedienten Unixbenutzer muß dies wie ein wahr gewordener Traum anmuten.

## Mitstreiter gesucht ... RMS und sein freies Unix

Jetzt denken Sie so langsam, daß nach dieser schon etwas langatmigen Einleitung doch sicher bald der Zeitpunkt kommt, an dem Linus Torvalds seinen tollen neuen 386SX auspackt und mit dem Programmieren von Linux beginnt. Weit gefehlt... . Technisch gesehen fängt die Linux-Geschichte auch nicht (erst) mit Linux an, zumindest dann nicht, wenn man Linux als etwas sieht, das über den bloßen Kernel hinausgeht.

Die Linux-Geschichte beginnt im September 1983 mit Richard Stallman. Stallman war Wissenschaftler am Massachusetts Institute for Technology (MIT) in Cambridge, Massachusetts, als er die erste Ankündigung von GNU (siehe Box, "Here's the GNUs") machte. GNU sollte später zur Free Software Foundation werden. Zu dieser Zeit war Stallman schon bekannt als der Autor eines Texteditors namens Emacs. Lesern mit einem weit in die Vergangenheit reichenden Unixgedächtnis werden die Heiligen Kriege zwischen vi- und Emacs-Benutzern sicher noch gut in Erinnerung sein.

Here's the GNUs - RMS kündigt sein freies Unix-System an

Tue, 27-Sep-83 12:35:59 EST

Free Unix!

Ab Thanksgiving werde ich ein freies Unix-kompatibles Softwaresystem schreiben, das GNU heißen wird (GNU für GNU's Not Unix). Ich werde es jedem umsonst(1) weitergeben, der es gebrauchen kann. Beiträge in Form von Zeit, Geld, Programmen und Ausrüstung sind dringend nötig.

Zuerstmal wird GNU ein Kernel plus all diejenigen Anwendungen sein, die es zum Schreiben und Laufenlassen von C Programmen braucht: Editor, Shell, C Compiler, Linker, Assembler und noch ein paar andere Dinge. Danach werden wir einen Textformatierer, YACC, ein Empire Spiel, eine Tabellenkalkulation und Hunderte anderer Sachen dazutun. Am Ende können wir hoffentlich alles das bieten, womit ein normales Unix-System aufwarten kann, und noch mehr nützliche Dinge wie Onlinedokumentation und gedruckte Dokumentation.

GNU wird mit Unixprogrammen laufen, aber es wird nicht identisch mit Unix sein. Wir werden alle Verbesserungen einbauen, die uns nach unseren Erfahrungen mit anderen Betriebssystemen nötig erscheinen.

Unter <http://www.gnu.org/gnu/initial-announcement.html> können Sie das Announcement in voller Länge nachlesen.

Die Komponenten eines freien Betriebssystems zu schreiben, braucht seine Zeit. Einer der wichtigsten Bestandteile des ursprünglichen GNU Systems, der Betriebssystemkernel, bekannt unter dem Namen Hurd, erblickte erst in jüngerer Zeit das Licht der Welt (von Experimentiersystemen abgesehen). Wie auch immer, Richard Stallman machte eine Fülle von Anwendungen verfügbar, ohne die ein beliebiges Betriebssystem völlig nutzlos sein würde. Von Emacs, dem Texteditor, einmal abgesehen, ist es ein grosses Verdienst Richard Stallmans, den GNU C Compiler (oder auch GCC) entwickelt zu haben. Eines der höchsten Designziele war es, den GCC so portabel wie möglich zu entwickeln. So gibt es heute für alle Betriebssysteme unter der Sonne eigene GCC Versionen. Es gibt auch für viele andere Programmiersprachen wie C++, Pascal und Fortran eigene GNU C Compiler. Die GNU Anwendungen waren enorm wichtig für die Entwicklung von Linux.

## **GNU, freie Software und Open Source**

Trotzdem ist es möglicherweise Stallmans größtes Verdienst, die GNU General Public License (GPL) geschaffen zu haben. Per Definition ist alle GNU Software unter dieser Lizenz oder der weniger restriktiven LGPL. In Kurzform besagt die GPL, daß wann immer Sie Änderungen an einer Software (oder Teilen davon) vornehmen, die unter der GPL steht, Sie diese Änderungen wiederum unter der GPL zur Verfügung stellen müssen. Wenn Sie Software verkaufen, die unter der GPL steht, müssen Sie den kompletten Quellcode all denjenigen verfügbar machen, die dazu Zugang haben wollen. Darüberhinaus hat jeder, der GPL Software benutzt, auch wieder das gleiche Recht, diese weiterzuvertreiben oder zu verändern. Die GPL ist so ausgelegt, daß Sie immer die Freiheit behalten, mit der Software alles anzustellen, was Sie wollen. Aus diesem Grund wird sie auch oft als "Copyleft" bezeichnet.

Richard Stallman und die GPL lieferten den Anstoß für eine völlig neue Kultur der Softwareentwicklung, die die Free Software oder Open Source Bewegung genannt wird. Jeder, der möchte und dazu fähig ist, kann seine Änderungen in GPL Software einbringen. Und in der Tat hat, wie die letzten 17 Jahre gezeigt haben, die GPL zu einem rasanten evolutionären Fortschritt in der Verbesserung von GPL Software geführt.

An dieser Stelle ist wichtig festzuhalten, daß Linux nicht das erste Softwareprojekt ist, das unter dem Paradigma von Open Source steht, obwohl es möglicherweise das bis heute erfolgreichste von allen ist.

## ... bloß so ein Hobby ...

Jetzt endlich nähern wir uns den Anfängen. Es ist 1991. Als Versuch, eigene Erfahrungen beim Betriebssystemdesign zu sammeln, begann Linus Torvalds, Informatikstudent an der Universität von Helsinki, Finnland, mit der Arbeit an einem neuen Betriebssystem. Es würde später nach seinem Begründer Linux getauft werden. Die ersten Releases hießen aber FREAX, nicht Linux. Im Kernel/Makefile der Version 0.11 und anderem Code findet sich immer noch dieser Name. Die allerersten Entwicklungsschritte von Linux fanden auf einem ziemlich unspektakulär bestückten 386SX statt. Linus' erstes Posting in einer Newsgroup über Linux ging am 25. August 1991 an comp.os.minix (siehe Box, "Linus stellt Linux vor").

### Linus stellt Linux vor

Hallo alle da draußen, die Ihr Minix benutzt,

ich bin dabei, ein (freies) Betriebssystem für 386(486) AT Klone zu machen (bloß so ein Hobby, es wird nicht riesig und professionell wie GNU werden). Das gärt schon seit April so vor sich hin, und so langsam beginnt es fertig zu werden. Ich hätte gern von Euch Feedback, was Leute an Minix mögen oder auch nicht mögen, weil mein Betriebssystem ihm irgendwie ähnelt (unter anderem gleicht es ihm im physikalischen Layout des Dateisystems (aus praktischen Gründen)).

Zur Zeit habe ich bash(1.08) und gcc(1.40) portiert, und die Sachen scheinen zu laufen. Das deutet darauf hin, daß ich in den nächsten paar Monaten da etwas Präsentables auf die Beine bringen werde. Ich würde gerne wissen, welche Features sich die meisten Leute wünschen. Alle Vorschläge sind herzlich willkommen, aber ich werde bestimmt nicht versprechen, daß ich sie auch implementieren werde :-)

Linus (torvalds@kruuna.helsinki.fi)

P.S.: Ja - es ist kein Minix Code drin und es hat ein Multi-Threaded Dateisystem. Es ist NICHT portierbar (benutzt 386 Switching Code), und es wird wohl auch nie etwas anderes als AT-Festplatten unterstützen, weil ich nämlich nichts anderes habe :(

Zur Zeit dieses Postings, mit Version 0.01, wies Linux oder FREAX, das auf ziemlich unspektakulärer Hardware entwickelt wurde, noch sehr wenig Funktionalität auf.

In einem anderen Posting vom 26. August 1991, verpflichtet sich Linus erstmals der GNU Lizenz gegenüber: "... Auch dann wird es wahrscheinlich nicht viel mehr können als Minix, und in mancher Hinsicht eher noch viel weniger. Aber trotzdem wird es frei sein (wahrscheinlich unter der GPL oder so etwas in der Art)... "

Die Tatsache, daß Linus seinen Code frei über das Internet verfügbar machte, ist enorm wichtig für die weitere Entwicklung von Linux. Viele Leute gesellten sich mit Nachfragen nach bestimmten Features oder noch besser mit selbst implementierten Erweiterungen oder Features hinzu. Es muß angemerkt werden, daß 1991 der Gebrauch des Internets noch nicht sehr weit verbreitet war. Die Beiträge kamen also von Personen, die selbst relativ stark technisch "angehaucht" waren.

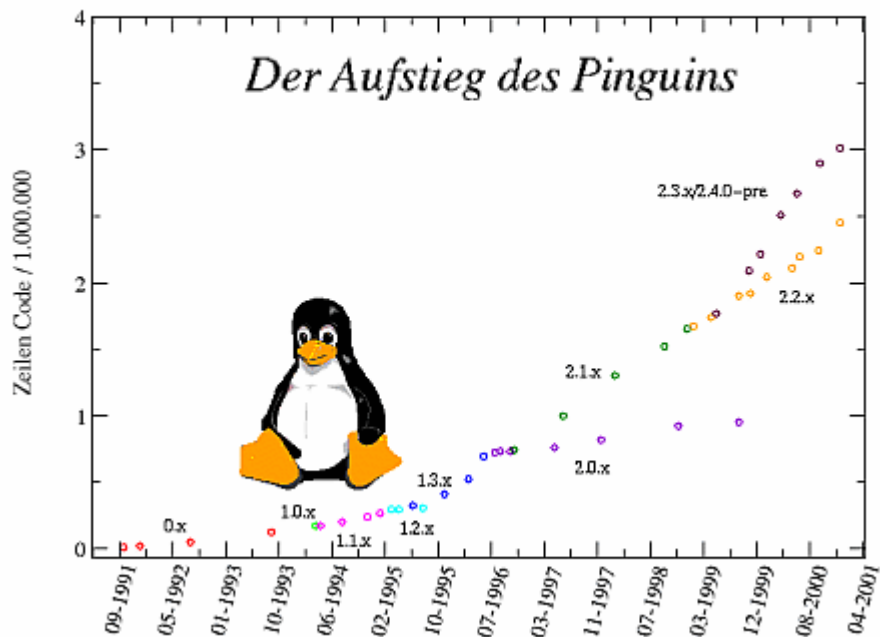
Nach zwei weiteren Jahren erreichte Linux am 16. April 1994 die Versionsnummer 1.0. Das bedeutet nicht, daß Linux vor dieser Zeit nicht benutzbar gewesen wäre. Viele studierten zu dieser Zeit an der Universität und setzten Linux für alle möglichen Arten von Aufgaben ein. In Tabelle 1 sind die

wichtigsten Eckpunkte der Entwicklung von Version 0.01 bis 1.0 stark komprimiert dargestellt.

Table 1. Die Entwicklung von 0.01 auf 1.0

Version	Datum	Kommentare
0.01	09/1991	keine (Binär-)Programme verfügbar, einige Gerätetreiber und ein Festplattentreiber
0.03	26/10/1991	für benutzbar befunden, Shell verfügbar, C Compiler und ein paar Anwendungen
0.12	05/01/1991	Die erste unter der GPL vertriebene Version
0.96	04/1992	Erste Version mit funktionierendem X Window System
0.99.14	12/1993	Die 0.99er Serie hatte eine Menge Sub-Versionen, je näher die Version 1.0 rückte
1.0	16/04/1994	Nach mehr als zwei Jahren Entwicklungsarbeit Release der Version 1.0

Mit Version 1.0 wurde die Entwicklung des Linux Kernels in zwei Linien aufgespalten. Gerade Versionsnummern (1.0, 1.2, ...) bezeichnen stabile, gebrauchsfertige Kernel. Ungerade Versionsnummern (1.1, 1.3, ...) bezeichnen Entwicklerkernel und sind allein für Entwicklungszwecke gedacht. Sobald alle gewünschten Features in eine Serie von Entwicklerkernen eingeflossen sind, wird die Endversion einem Code Freeze unterzogen und schließlich in einen ersten Kernel einer stabilen Serie umbenannt. Normalerweise sind in einer Serie stabiler Kernel nur Bugfixes erlaubt, wenn auch manchmal wichtige neue Features trotzdem aus der Entwicklerserie zurückportiert werden.



Für die verschiedenen Kernelversionen ist jeweils die Anzahl der Codezeilen angegeben. Offensichtlich nimmt das Codevolumen der stabilen Kernelversionen (2.0, 2.2) sehr viel weniger stark zu als das der Entwicklerkernel (2.1, 2.3). Die aktuelle Kernelversion 2.4 hat mehr als drei Millionen Zeilen, im

Vergleich dazu hatte Version 0.01 8400. Kernel 1.0 mit 170.000 Zeilen war ein großer Fortschritt.

Von Beginn an hing die Entwicklung von Linux von vielen anderen Einzelpersonen ab, die ihre Zeit, ihr Können und bemerkenswerte Anstrengungen in dieses Projekt investierten (siehe Box, " Einige Schlüsselentwickler").

#### Einige Schlüsselentwickler

---

Werner Almesberger Treiber für Diskettenlaufwerke

Theodore Ts'o Ext2 Dateisystem, Libraries, Kernel Memory Allocator, viele Beiträge, um Linux POSIX kompatibel zu machen

Donald Becker Netzwerktreiber (später auch: Beowulf)

Olaf Kirch Das Linux-Netzwerker Buch, NFS Code

Alan Cox Netzwerk, eine Menge Entwicklungsarbeit am Kernel, frühe SMP Implementierung, ... Alan Cox ist Maintainer der 2.2er Kernelserie und mittlerweile einer der Schlüsselentwickler des Kernels.

---

Natürlich haben diese Entwickler noch viel mehr beigetragen als die genannten Beispiele, und natürlich gibt es auch eine riesige Menge anderer Entwickler, die nicht genannt wurden und sehr wertvolle Beiträge leisteten.

Im Laufe der Zeit wurden viele neue Features in den Kernel integriert. Die SMP Fähigkeit (Symmetric Multiprocessing) wurde mit Kernel 2.0 eingeführt. Sie war aber schon vorher auf experimenteller Basis verfügbar. Peter McDonald führte zum ersten Mal ladbare Module für einen 0.99er Kernel ein. Die heute gebräuchliche Implementierung ist allerdings völlig unterschiedlich von dieser ursprünglichen Version. Mit Kernel 1.2 (6.3.1995) wurde die Verfügbarkeit von Linux formal auf Alpha, Sparc und MIPS Prozessoren ausgeweitet und so die starke Abhängigkeit des Kernels von x86 Prozessoren in älteren Versionen überwunden. Heute ist Version 2.4 für jede wichtige Prozessorplattform verfügbar, angefangen am unteren Ende bei den Motorola 68000 Prozessoren bis zum IBM S/390 Mainframe am anderen Ende der Skala.

Aus Benutzersicht sind allerdings die neuen Treiber die größte und wichtigste Neuerung. Zu Beginn war da nicht viel mehr als ein Treiber für ein Standard IDE Laufwerk, spätere Versionen umfassen die Unterstützung für sämtliche Arten von Hardware. Entscheidend für diese Entwicklung war die breite Unterstützung von Linux durch freiwillige Entwickler über das gesamte Internet verstreut, sowie die Bereitschaft der Hardwarehersteller, die Spezifikationen ihrer Hardware öffentlich zugänglich zu machen. Nicht alle Hersteller sind so entgegenkommend. Trotzdem braucht die Hardwareunterstützung für Linux keinen Vergleich zu scheuen. Kernel 2.4 bringt sogar die Unterstützung von USB und Firewire Treibern. Trotz allem wird die beständige Unterstützung seitens der Hardwarehersteller enorm wichtig für das zukünftige Wachstum von Linux bleiben.

*Den zweiten Teil dieser Serie wird Rüdiger einem Rückblick auf die ersten Linux Distributionen, der Debatte um Standards und dem Aufkommen einiger Anwendungen widmen. Rüdiger Berlich ist Managing Director von [SuSE Linux UK](#).*

*Diese Artikelfolge erscheint mit freundlicher Genehmigung des britischen "[Linux-User](#)" Magazins. Die Rechte an dieser Veröffentlichung liegen beim Autor.*

