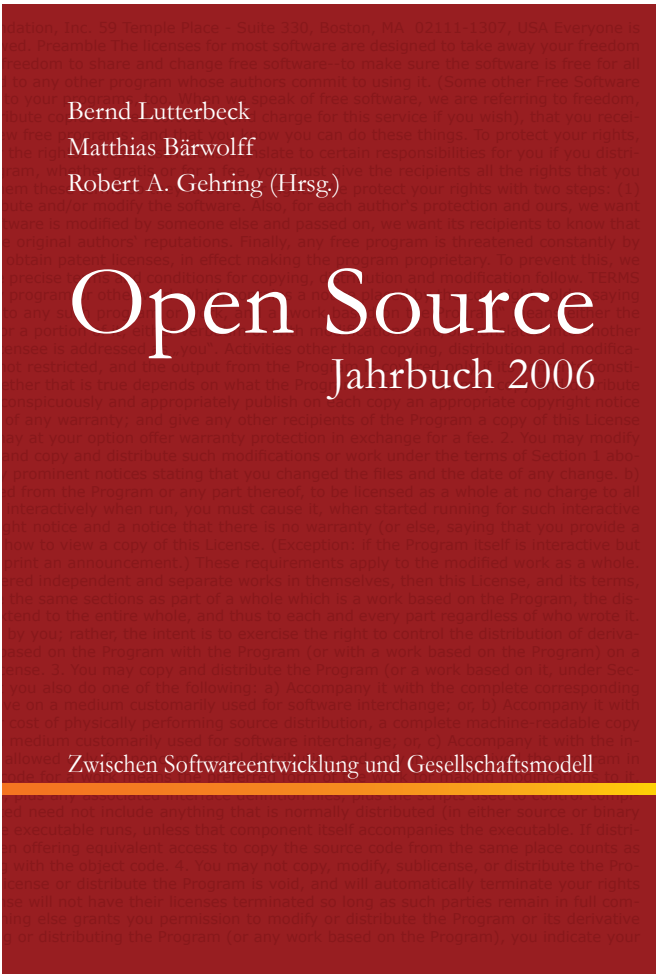
Dieser Artikel ist Teil des

Open Source Jahrbuchs 2006



erhältlich unter www.opensourcejahrbuch.de.

Die komplette Ausgabe enthält viele weitere interessante Artikel. Lob und Kritik zu diesem Artikel sowie weitere Anregungen können Sie uns einfach und unkompliziert mitteilen per E-Mail oder auf www.opensourcejahrbuch.de/feedback/.

Erfolgreich mit Open Source – Das Red-Hat-Open-Source-Geschäftsmodell

Werner Knoblich*

Der folgende Text stellt das Red-Hat-Geschäftsmodell vor dem Hintergrund der Entstehung von Open-Source-Geschäftsmodellen in den späten 90er Jahren vor. Red Hat ist heute der einzige unabhängige Open-Source-Anbieter, der durch das Angebot eines Standardproduktes in Form eines Abonnements – der Subskription – profitabel ist. Unternehmen schenken Red Hat ihr Vertrauen aufgrund der Zuverlässigkeit und Stabilität des Produkts, nicht wegen technischer Features.

Schlüsselwörter: Open-Source-Geschäftsmodell · Subskription · OSS-Lizenz

1 Open Source als Lizenzentscheidung von Red Hat

Abgegrenzt wird Open-Source-Software (OSS) von proprietärer Software – Software, deren Lizenzbedingungen dem Anwender nicht die genannten Rechte einräumen: Während das immer noch vorherrschende Modell proprietärer Software im Allgemeinen darauf abzielt, die Freiheiten des Anwenders im Umgang mit der Software auf ein Minimum zu beschränken, erweitert OSS den Handlungsspielraum der Nutzer durch seine spezifische Lizenzierung erheblich.

Innerhalb des durchaus heterogenen Spektrums der OSS-Lizenzen sind drei wesentliche Lizenzfamilien zu unterscheiden:

– So genannte „Copyleft"-Lizenzen, welche die OSS-Rechte mit der Verpflichtung verbinden, bei der Weitergabe der Software stets auch dieselben Rechte weiterzugeben. Diese Lizenzen garantieren die freie Verfügbarkeit der Software

* Werner Knoblich ist *Red Hat Vice President Europe, Middle East and Africa (EMEA)*, verantwortlich für Koordinierung und Ausbau des Red-Hat-Geschäfts in Europa, dem mittleren Osten und Afrika.

und schützen vor einer Vereinnahmung in proprietären Produkten. Die wichtigste Lizenz dieser Kategorie – und insgesamt die meistgenutzte OSS-Lizenz – ist die *GNU General Public License (GPL)*[1].

- Liberale Lizenzen wie die (modifizierte) BSD-Lizenz, die MIT- oder Apache–Lizenz, die über die Rechte der GPL hinausgehen, und es erlauben, den Code in proprietären Produkten zu verwenden.
- GPL-inkompatible, limitierte Lizenzen, die eine Mischung des Codes mit Copyleft-lizenziertem Code verbieten. Diese Lizenzen werden meist von einzelnen Herstellern verwendet, welche zwar Programmcode als OSS freigeben, dabei jedoch die Exklusivität wahren und verhindern wollen, dass andere OSS-Projekte von dem Code profitieren. Aktuell die bekanntesten Beispiele dürften hier die Protected-BSD-Lizenz oder SUNs CDDL sein.

Zwischen kommerzieller Software und OSS gibt es keinen immanenten Widerspruch per se. Ganz im Gegenteil: OSS kann kommerziell entwickelt, kommerziell vertrieben und kommerziell eingesetzt werden. OSS ist lediglich im Allgemeinen frei von Lizenzgebühren, da insbesondere aufgrund des Rechts der Weiterverbreitung jede Bemessungsgrundlage fehlt. Dies steht jedoch nicht anderen kommerziellen Verwertungsmodellen entgegen. Nicht zuletzt ist die GPL die favorisierte Lizenz von Red Hat. Das heißt alle Produkte von Red Hat, auch Red Hat Enterprise Linux (RHEL), werden unter einer GPL vertrieben.

2 Die Vorteile des Open-Source-Entwicklungsmodells

Ein entscheidender Aspekt von OSS ist das kooperative Entwicklungsmodell, gestützt durch die insbesondere mit GPL- und BSD-artigen Lizenzmodellen verbundenen Freiheiten und die Kommunikationsmöglichkeiten des Internets. Nicht mehr nur einzelne Unternehmen oder Entwickler arbeiten an Programmcode für einen einzelnen, klar definierten Zweck. Vielmehr findet ein aktiver Austausch innerhalb einzelner Projekte oder zwischen Projekten statt. Daraus folgen diverse strukturelle Vorteile gegenüber dem proprietären Entwicklungsmodell:

Innovationsgeschwindigkeit Das kollaborative Entwicklungsmodell, in dem zehntausende von Ingenieuren und Entwicklern rund um den Globus in der Open–Source-Entwicklungsgemeinschaft arbeiten, gewährleistet kürzeste Entwicklungszyklen (siehe hierzu auch von Hippel 2005).

Qualität und Sicherheit Durch die große Gemeinschaft wird der Quellcode von Open–Source-Software von viel mehr Entwicklern untersucht und getestet, als dies bei proprietärer Software der Fall sein kann. Bugs und Sicherheitslücken werden frühzeitig aufgedeckt und behoben.

1 http://www.gnu.org/copyleft/gpl.html

Flexibilität Höhere Flexibilität und Anbieterunabhängigeit sind zwei der Hauptargumente, die für den Einsatz von Open-Source-Software sprechen.[2] So war beispielsweise die Möglichkeit, herstellerunabhängige Hardware und Anwendungssoftware einsetzen zu können, in den vergangenen Jahren die Triebfeder für den Erfolg von Linux.

Wahlfreiheit Unabhängigkeit von proprietären Hardware- und Softwarelösungen eines Herstellers bietet signifikantes Einsparungspotential und Zukunftsicherheit.

Nachhaltigkeit Da Open-Source-Software unabhängig von einem bestimmten Hersteller, und durch seine Offenheit für jeden einsehbar und modifizierbar ist, ist eine langfristige Nutzbarkeit gewährleistet.

Insbesondere die Wahlfreiheit des Kunden hat eine entscheidende strategische Dimension. Nicht nur die Kombination von Hardware, System- und Anwendungssoftware kann durch den Kunden kontrolliert werden, er kann sich in letzter Konsequenz auch von einzelnen Anbietern trennen, etwa wenn er mit deren Service nicht mehr zufrieden ist. Aber auch Sicherheit und Kosten sind bei der konkreten Produktentscheidung von großer Bedeutung, wobei die langfristigen Betriebskosten im Vordergrund stehen.

OSS wird heute auf breiter Basis eingesetzt. Am meisten verbreitet ist der Apache Webserver mit einem aktuellen Marktanteil von rund 70 Prozent.[3] Stark verbreitet ist Open Source jedoch auch in der Softwareentwicklung, insbesondere im Java-Umfeld, bei Hilfs-Bibliotheken und Toolkits sowie im wissenschaftlich-technischen Bereich. Ökonomisch am interessantesten ist derzeit jedoch der Bereich der Betriebssysteme, und hierbei insbesondere Linux, da es sich als vollständiges System im Markt etablieren konnte und zum Katalysator für die Entstehung verschiedener Open–Source-Geschäftsmodelle wurde.

3 Wie lässt sich Geld mit Freier Software verdienen?

Die Frage, wie sich auf Basis von OSS tragfähige Geschäftsmodelle etablieren lassen, wird spätestens seit dem Internet- und „New Economy"-Boom in der zweiten Hälfte der 90er Jahre intensiv diskutiert – und erprobt. Das erste klar kommerzielle Unternehmen mit einem vollständigen Open-Source-Geschäftsmodell war vermutlich Cygnus Solutions, gegründet 1989, seit Ende 1999 Teil von Red Hat.

In der Folge entwickelten sich verschiedene Ansätze, die sich trotz einiger Überschneidungen in eine Reihe von Kategorien[4] einteilen lassen:

2 Siehe etwa eine Studie von Actuate: „Finanzdienstleister sind Open Source-freundlich", http://www.actuate.com/ger/unternehmen/presse/PressRelease.asp?ArticleId=9398.

3 Siehe hierzu die Statistiken unter http://news.netcraft.com/archives/web_server_survey.html.

4 Eine interessante Übersicht der klassischen Modelle mit einer etwas anderen Einteilung findet sich auch in Leiteritz (2004).

Distributoren Das ursprüngliche Kerngeschäft der Distributoren bestand im Vertrieb von installierbaren Linux-Distributionen auf Datenträgern inklusive Dokumentation und zeitlich befristeten Support-Leistungen gegen eine einmalige Gebühr. In der Distribution werden dabei Komponenten aus einer Vielzahl unterschiedlicher Upstream-Projekte zusammengefasst.

Appliances Manche Anbieter entwickeln „Appliances", die eine Kombination von Hardware, Betriebssystem und Software bilden. Das können neben Server–Appliances z. B. auch Set-Top-Geräte oder DVD-Player sein, bei denen der Kunde z. B. Linux als das eigentliche Betriebssystem nicht mehr wahrnimmt.

Dienstleister Consulting-Firmen, die sich auf Support und Services rund um Open–Source-Software spezialisiert haben, selbst aber kein eigenes Produkt, wie z. B. eine eigene Linux Distribution, anbieten.

Hybridmodelle Anbieter, die verschiedene der oben aufgeführten Modelle vereinen, z. B. eine eigene Distribution anbieten, aber auch Services und Support für diese Distribution und andere Open-Source-Produkte.

Sekundärmärkte Bei dieser Variante des Geschäftsmodells wird die Open-Source-Software komplett kostenlos abgegeben. Das heißt somit, dass sich der Anbieter einen alternativen Markt für seinen finanziellen Erfolg kreieren muss.

Hardware-Enablement Dieses Geschäftsmodell fokussiert sich primär auf den Vertrieb von PC-Hardware, die direkt zum Verkauf mit einem Open-Source-Betriebssystem als so genannter „Preload" ausgestattet wird. Weitere OSS-Applikationen können dem Angebot beigefügt sein.

Loss-Leader Hierbei wird das Open-Source-Produkt als sehr günstiges Lockangebot verwendet, um potentielle Käufer anzuziehen und im Zuge dessen ein anderes, meist hochpreisiges Produkt mitzuverkaufen.

Hybrid-Lizenzen bieten nur einige der Vorteile des OSS-Modells – vor allem verweigern sie dem Kunden die Wahlfreiheit. Sekundärmärkte, Loss-Leader und Hardware–Enablement folgen naturgemäß eigenen Gesetzmäßigkeiten. Vor allem die Distributoren, Appliances und Dienstleister sind hier also von Interesse, da es sich dabei um originäre OSS-Geschäftsmodelle handelt. Diese erfuhren in der Hochphase des „New Economy"-Booms die besondere Aufmerksamkeit von Risikokapitalgebern.

4 Erfolg und Misserfolg von Open-Source-Firmen vor und nach dem Internet-Boom

Als erstes Open-Source-Unternehmen ging Red Hat im Jahre 1999 erfolgreich an die Börse. Es folgte VA Linux Systems, die einen der erfolgreichsten Börsengänge aller

Zeiten durchführte. Hier ist allerdings anzumerken, dass es sich bei VAs Geschäftsmodell eigentlich um ein Hardware-Enablement-Modell handelte. Linux und Open Source waren streng genommen nur ein Marketinginstrument und Abgrenzungskriterium. Auch Dienstleister wie LinuxCare oder Mission Critical Linux wurden massiv mit Risikokapital ausgestattet. In Europa wurde in Unternehmen wie die Distributoren SuSE und Mandrake oder beispielsweise die Service-Unternehmen ID-Pro, Innominate, Alcove oder Idealix investiert.

Allerdings zeigte sich mit dem Ende des New-Economy-Booms in den Jahren 2000 und 2001, dass die Geschäftsmodelle und Strategien vieler Open-Source-Unternehmen nicht ausgereift waren. VA Linux scheiterte an dem Versuch, sich als weltweiter Hardwarehersteller neben Dell, Sun, IBM, HP und damals noch Compaq zu positionieren.[5] SuSE gelang es nie, schwarze Zahlen zu schreiben, rettete sich von VC-Runde zu VC-Runde, verbunden mit entsprechenden Management-Wechseln und wurde schließlich 2003 von Novell übernommen. Mandrake – nach Fusion mit dem brasilianischen Distributor Connectiva unter dem Namen Mandriva firmierend – überstand zwei Insolvenzen und mehrere grundlegende Strategieänderungen, scheint jedoch inzwischen stabilisiert. LinuxCare, ID-Pro, Innominate und zahllose weitere VC-finanzierte Linux-Dienstleister scheiterten an zu optimistischen Marktprognosen. Sie machten den Weg frei für kleinere, ökonomisch flexiblere OSS-Dienstleister. Von den „großen" OSS-Anbietern blieb ausschließlich Red Hat unabhängig.

5 Auf der Suche nach einem profitablen Geschäftsmodell

Für die Distributoren, die traditionell einen durchaus nennenswerten Anteil der OSS–Entwicklung tragen, stellte sich das Problem, dass in Zeiten von Breitband-Internet der Mehrwert eines Linux auf einem Datenträger relativ gering ist. Aufgrund der schnellen Entwicklung in den vorgelagerten Projekten veralten Produkte relativ schnell und Umsätze sind nur über häufige Releases mit möglichst vielen Features möglich. Durch diese Strategie trat jedoch schnell ein gewisser Sättigungseffekt ein, begleitet von der Tendenz, schlicht online die neuen Paketversionen herunterzuladen. Zudem erfordert der Vertrieb im „Boxen"-Geschäft einen nicht unerheblichen Aufwand. Ein echtes tragfähiges Geschäftsmodell ließ sich daraus nicht entwickeln – so waren Red Hats profitable Geschäftsbereiche zu dieser Zeit vor allem Consulting und Training.

Eine Gemeinsamkeit von Distributoren, Dienstleistern und Appliance-Anbietern ist die starke Dienstleistungsorientierung. So waren die Distributoren in aller Regel auch Dienstleister und viele Dienstleister hatten (und haben) eigene Appliance–Angebote. Ein auffälliges Charakteristikum der Dienstleistungs-Angebote war stets ein hohes Maß an projektspezifischer Individualisierung: angepasste Linux-Stacks, individuelle Supportverträge etc. Typisch für die Appliance-Angebote – ob inklusi-

5 Heute betreibt die Firma noch das berühmte Web-Portal Slashdot.org.

ve Hardware oder reine Software-Appliances – war das vollständige Angebot eines durchgängigen, individuellen Software-Stacks inklusive des Basisbetriebssystems.

Ein zentrales Problem dieser Ansätze ist vor allem fehlende Skalierbarkeit des Geschäftsmodells. Eine individualisierte Dienstleistung auf Basis von Linux unterscheidet sich letztlich nicht maßgeblich von anderen Dienstleistungsangeboten im IT-Bereich. Umsätze skalieren direkt über die Anzahl der Mitarbeiter. Damit unterliegen diese Firmen letztlich den gleichen Marktbedingungen wie Dienstleister im proprietären Markt, die sich jedoch auf einem standardisierten Software-Stack bewegen können.

Der OSS-Dienstleister muss sich einem weiteren Problem stellen: Die Möglichkeit, aus dem kommerziellen Geschäft heraus aktiv die Weiterentwicklung der OSS-Basis voranzutreiben, ist stark limitiert. Letztendlich lassen sich stets nur die konkreten Projektziele implementieren. Ein strategischer Einfluss ist relativ begrenzt und gelingt höchstens in klar abgegrenzten Kernkompetenz-Bereichen. So beschränken sich diese Dienstleister in der Breite meist auf reines Customizing der Software aus der Community. Und auch Appliance-Anbieter, welche über das individualisierte Direktgeschäft hinaus einen Vertrieb etablieren, sind durch die Pflege eines eigenen, durchgängigen Software-Stacks belastet. Eine echte, strategische Weiterentwicklung der Open-Source-Basis findet nach unserer Erfahrung kaum statt.

Auch auf der Kundenseite kommen mit wachsender Komplexität der eingesetzten OSS-Systeme im professionellen Einsatz weitere Herausforderungen hinzu. So ist ein Nachteil der hohen Innovationsgeschwindigkeit von OSS, dass Bugs und Sicherheitsprobleme vor allem in kleineren, vorgelagerten Projekten ausschließlich in der jeweils neuesten Version behoben werden. Neue Versionen bedeuten jedoch meist auch neue Features und damit neue Fehler. Zudem sind sie oft mit Änderungen in der Laufzeitumgebung und der Konfiguration verbunden, was den Aufwand ihrer Installation in die Höhe treibt. In unternehmenskritischen Umgebungen können Änderungen oder Upgrades daher teuer, kostenintensiv und vor allem relativ schwer abschätzbar sein. Dies kann dazu führen, dass die entsprechenden Entscheider es im Zweifel vorziehen, Sicherheitslücken in Kauf zu nehmen und auf Einspielung von Updates zu verzichten.

Verstärkt wurden diese Probleme noch durch das Geschäftsmodell der Distributoren. Da der Verkauf neuer CDs ihre primäre Einnahmequelle darstellte, versuchten sie, möglichst häufig neue Versionen der Distributionen zu platzieren, die möglichst viele neue Funktionalitäten enthielten. In der Folge gab es zeitweise allein von den vier oder fünf größeren Distributoren bis zu 16 unterschiedliche Kombinationen von Linux-Betriebssystem, Laufzeitumgebung und Applikations-Stack pro Jahr. Nicht zuletzt wegen der Zertifizierung unterschiedlicher Distributionen und Versionen seitens verschiedener Softwarehersteller hatten Kunden damit je nach Zeitpunkt der Inbetriebnahme ihrer Systeme eine ansehnliche Anzahl unterschiedlicher Linux-Varianten in Betrieb, was wiederum zu erhöhten Betriebskosten führte. Diese Vielfalt von Installationen konnte dazu führen, dass Kunden in die Situation gerieten, Sicherheitsupdates aus vorgelagerten OSS-Projekten nicht installieren zu können, weil verschiedene Dis-

tributoren unterschiedliche Versionen zertifizierten.

Aufgrund der inneren Abhängigkeiten der verschiedenen aufeinander aufbauenden Komponenten eines Software-Stacks, auch unter Linux, wird dieses Problem noch weiter verstärkt. Wegen der schnellen Entwicklung erfordern einzelne Versionen von Anwendungen und Bibliotheken ihrerseits bestimmte Versionen der darunter liegenden Bibliotheken. Erfordert nun ein Bugfix in einer Komponente des Stacks die Installation einer neuen Version, kann dies im ungünstigsten Fall die Installation eines vollständig neuen Software-Stacks nach sich ziehen.

Aufgrund der Fokussierung des Dienstleistungsmarktes auf individualisierte Lösungen schließlich waren viele Installationen beim Kunden echte „Einzelkunstwerke", bei denen der Linux-Kernel individuell übersetzt wurde, einzelne Pakete aus verschiedenen Quellen stammen und durch Patches modifiziert wurden. Solche Nicht-Standard-Umgebungen sind nur mit entsprechendem Personalaufwand und Know-how wartbar. Bereits die Relevanzprüfung für Sicherheitsupdates aus der vorgelagerten Community ist dabei eine nicht-triviale Problemstellung, da zu deren Beurteilung intime Kenntnisse der einzelnen installierten Pakete und Patches erforderlich sind.

Die Probleme traditioneller OSS-Geschäftsmodelle auf Kundenseite lassen sich wie folgt zusammenfassen:

- Häufige Änderungen der Programmierschnittstellen (API) und der binären Laufzeitumgebung (ABI) in der vorgelagerten Entwicklung,[6]
- Bugfixes nur in neuen Versionen,
- Zu schneller und häufiger Versionswechsel vieler OSS-Projekte für sinnvolle Zertifizierungen,
- Fehlendes durchgängiges Design und daher Redundanzen,
- Fehlen einer klar definierten Plattform,
- Eine zu große Anzahl an Individuallösungen.

Diese Probleme behinderten lange Zeit die Adaption von Linux im Bereich der professionellen IT und gefährdeten nicht zuletzt den langfristigen Erfolg. Während der IT-Betrieb kommerzieller Anwender im Allgemeinen langfristige Stabilität erfordert, war das Linux-Angebot geprägt von häufigen Änderungen und einer unsicheren ökonomischen Basis der Anbieter.

6 Neue Wege – das erfolgreiche Red-Hat-Geschäftsmodell

Mit einem neuen Geschäftsmodell griff Red Hat diese Herausforderung im Jahre 2002 auf. Red Hat verstand, dass sowohl Anbieter von Software (Independent Software Ven-

6 Eine Änderung des *Application Binary Interfaces (ABI)* hat zur Folge, dass Programme neu kompiliert werden müssen. Eine Änderung des *Application Programming Interfaces (API)* bedeutet, dass zudem der Quelltext des Programms verändert werden muss.

dors) als auch professionelle Anwender nicht einfach nur immer neue Linux-Versionen wollten, sondern eine langfristige Pflege ihrer bestehenden Umgebung.

Mit dem Red Hat Advanced Server, wenig später ausdifferenziert und umbenannt in Red Hat Enterprise Linux (RHEL), brachte Red Hat im Jahre 2002 ein Angebot auf den Markt, das diese Herausforderungen löste und gleichzeitig dem Kunden alle Vorteile des Open-Source-Modells zugänglich machte.

Im Kern besteht das Modell Red Hats aus einer definierten und qualitätsgesicherten Linux-Plattform in Verbindung mit einem Abonnement für die Bereitstellung von Sicherheitskorrekturen, Bugfixes und Feature-Aktualisierungen (Updates). Außerdem hat der Kunde Zugang zu neuen Major-Releases (Upgrades). Hinzu kommen interaktiver Support und eine juristische Absicherung gegen etwaige Urheberrechtsansprüche Dritter. Diese Plattform hat also effektiv den Charakter eines klassischen Standardproduktes und kann daher Hardware- und Software-Herstellern als Zertifizierungsreferenz dienen. Der entscheidende Punkt des Enterprise-Linux-Modells ist jedoch, dass jedes Major-Release für insgesamt sieben Jahre gepflegt wird und dass Red Hat sowohl die APIs als auch die Laufzeitumgebung, das ABI, stabil hält.

Red Hat erreicht diese ABI-Stabilität durch das Zurück-Portieren von Bugfixes und Sicherheitsupdates aus den vorgelagerten OSS-Projekten in die ursprünglich ausgelieferten Programmversionen. Das Einspielen neuer Versionen gefährdet damit nicht mehr die Funktionsfähigkeit der Produktivsysteme. Softwarehersteller können also schlicht eine Red-Hat-Enterprise-Linux-Version, wie z. B. das aktuelle RHEL4, zertifizieren, statt wie bisher bestimmte Kernel-Releases oder Bibliotheksversionen.

Neue Hardware wird mittels regelmäßiger Updates unterstützt. Neue Major-Releases werden ca. alle 18 bis 24 Monate veröffentlicht, wobei jeder Kunde im Rahmen seines Abonnements Zugang zu allen aktuell unterstützten Versionen hat. Daher ist jederzeit ein Upgrade auf eine neuere Version möglich – ohne zusätzliche Gebühren –, jedoch dank des siebenjährigen Produktlebenszyklus nur in sehr langen Abständen erforderlich. Neben dem Basisbetriebssystem bietet Red Hat noch eine Reihe von sogenannten „Layered Products" an, die auf RHEL aufbauen und nach demselben Prinzip angeboten werden: Red Hat Cluster Suite[7], Red Hat Global Filesystem[8], Red Hat Directory Server[9] und Red Hat Application Server[10].

7 Die Red Hat Cluster Suite bietet den Cluster-Manager, der durch Failover-Technologie eine hohe Verfügbarkeit der Anwendungen gewährleistet und die IP-Lastausgleichs-Technologie, die eine Lastverteilung ankommender IP-Netzwerkanforderungen auf Server in einer Serverfarm durchführt.

8 Das Red Hat Global File System (GFS) ist ein POSIX-fähiges Archivierungs- und Festplattenmanagementsystem für Cluster mit Storage Area Networks (SAN).

9 Der Red Hat Directory Server ist ein LDAP-basierter Server, der Funktionen wie Anwendungseinstellungen, Benutzerprofile, Gruppendaten, Regeln und Zugriffskontrollinformationen in einem zentralisierten, betriebssystemunabhängigen, netzwerkbasierenden Register zusammenfasst.

10 Der Red Hat Application Server ist eine Open-Source-Middleware-Plattform zwischen Betriebssystem und Applikationen und bietet die Grundlage zur Verbindung von Systemen und im Netzwerk verteilten Ressourcen. Er beinhaltet ein Runtime-System und dazugehörige Entwicklungs-Libraries, um Java-basierte Web-Anwendungen mit dynamischen Inhalten kreieren und implementieren zu können.

7 Das Entwicklungsmodell von Red Hat Enterprise Linux

Eine entscheidende Säule des Enterprise-Linux-Modells ist die klare Orientierung an der Open-Source-Community. RHEL und die übrigen Red-Hat-Produkte haben die positiven Eigenschaften eines Standardproduktes, bleiben dabei jedoch vollständig Open Source. Red Hat entwickelt neue Software konsequent in den einzelnen vorgelagerten Projekten. Auch eigene Initiativen werden so umgesetzt, dass sich eine eigenständige Entwicklergemeinde bilden kann. Damit hat Red Hat zwar keine Exklusivität auf Programmcode, kann jedoch einige der unbestrittenen Vorteile des Open-Source-Entwicklungsmodells für sich nutzen. Die Red-Hat-Produkte ihrerseits heben sich jedoch durch Stabilität und langfristige Vertrauenswürdigkeit vom Wettbewerb ab, nicht durch technische Features.

Innerhalb des von Red Hat initiierten Open-Source-Projekts Fedora erhalten die verschiedenen Softwarekomponenten aus dem Open-Source-Pool eine erste Form. So ist die Fedora-Core-Linux-Distribution die Entwicklungsplattform für RHEL. Fedora Core wird von Red Hat als Community-Distribution entwickelt, ist also gekennzeichnet durch relativ kurze Veröffentlichungszyklen, keine langfristige Maintenance und keine kommerziellen Dienstleistungen, sondern dafür durch neueste Technologie in einer grundlegend getesteten Distribution. Alle maßgeblichen Features eines RHEL–Release werden vorher innerhalb des Fedora-Projektes getestet. Fedora insgesamt beinhaltet neben der Basisdistribution weitere Projekte, wie z. B. den Directory-Server, und ist als von Red Hat finanzierte, jedoch unabhängige Stiftung organisiert.

Aufbauend auf Fedora Core entsteht RHEL, wobei die letzte Fedora Release als Alpha-Version dient. Damit ist bereits vor dem eigentlichen Produktentwicklungszyklus ein breiter Test sichergestellt. RHEL selbst ist ebenso wie alle anderen Kernprodukte von Red Hat Open Source. Die favorisierte Lizenz ist die GPL. Da es sich um ein Standardprodukt handelt, ist der Pflegeaufwand für den Kunden jedoch relativ gering. Die Bereitstellung von Sicherheits-Updates erfolgt über das Red Hat Network (RHN), eine zentrale Softwareverteilungsinfrastruktur, die – sofern der Kunde dieses Feature nutzt – eine automatische Relevanzprüfung durchführt und die für die Systeme des Kunden relevanten Updates direkt über einen gesicherten Kanal zur Verfügung stellt.

Eine wichtige Rolle innerhalb des Entwicklungsmodells von Red Hat nimmt das „Upstream-Commitment" ein, also Zusammenarbeit mit den vorgelagerten Communitys bei der Softwareentwicklung. Red Hat verteilt keinen Programmcode in seinen Produkten, der nicht von der jeweiligen vorgelagerten Community zumindest in einer späteren Version der Software akzeptiert wurde. Dadurch werden die oben angesprochenen Kompatibilitätsprobleme vermieden. Kunden bleiben in jedem Falle unabhängig in ihren IT-Entscheidungen, da es nicht zu einem herstellerspezifischen „Technologie Lock-In" kommen kann. Zudem wird die teure Wartung von exklusivem Code vermieden, da durch die Rückgabe von Code-Änderungen in die vorgelagerten Projekte die Konsistenz der Software gewahrt wird.

8 Fazit

Der Vorteil des Red-Hat-Enterprise-Linux-Modells für den Kunden ist vor dem Hintergrund der weiter oben dargelegten Open-Source-Herausforderungen die Einführung von architektonischer Stabilität in den Open-Source-Markt. Damit schafft es die Voraussetzung für einen breiten kommerziellen Erfolg von Linux und Open Source in unternehmenskritischen Anwendungen.

Dabei löst das Red-Hat-Enterprise-Linux-Modell auch die eingangs dargelegten ökonomischen Probleme. RHEL, ebenso wie die übrigen Red-Hat-Produkte, ist ein Standardprodukt. Es wird einmal entwickelt und kann prinzipiell beliebig oft verkauft werden. Es skaliert in seinen Kernkomponenten – Software und langfristige Maintenance – weitgehend unabhängig von der Mitarbeiterzahl und eignet sich dadurch, wie proprietäre Softwareprodukte, für den indirekten Vertrieb, ohne dabei die Vorteile des Open-Source-Entwicklungsmodells zu verlieren. Damit verschafft es Red Hat ein profitables Geschäftsmodell und wahrt die Unabhängigkeit, die für eine strategische Softwareentwicklung erforderlich ist.

Die Zahlen von Red Hat sprechen hier für sich. Entgegen dem Branchentrend hat Red Hat seit der Einführung von Enterprise Linux seine Umsätze drastisch gesteigert, operative Profitabilität erreicht und gleichzeitig die Mitarbeiterzahl verdoppelt. Im vierten Quartal 2005 konnte Red Hat den Umsatz um 44 % im Vergleich zum Vorjahresquartal auf 73,1 Mio. US-Dollar steigern. Der Umsatz mit RHEL-Subskriptionen stieg dabei um 54 % verglichen mit dem Vorjahresquartal.[11] Investoren konnten sich im Jahr 2005 über eine Kurssteigerung um 107 % und die Aufnahme der Aktie in den Nasdaq-100-Index freuen.

Mit den „Layered Products“[12] beweist Red Hat zudem, dass dieses „Enterprise Linux“-Modell nicht nur auf Betriebssysteme begrenzt ist. Es lässt sich als generelles Modell für kommerzielle und dennoch freie Software ausbauen.

Literatur

Leiteritz, R. (2004), Open-Source-Geschäftsmodelle, *in* B. Lutterbeck und R. A. Gehring (Hrsg.), ‘Open Source Jahrbuch 2004 – Zwischen Softwareentwicklung und Gesellschaftsmodell’, Lehmanns Media, Berlin, S. 139–170. http://www.opensourcejahrbuch.de/2004/ [02. Feb 2006].

von Hippel, E. (2005), „Anwender-Innovationsnetzwerke“: Hersteller entbehrlich, *in* B. Lutterbeck, R. A. Gehring und M. Bärwolff (Hrsg.), ‘Open Source Jahrbuch 2005 – Zwischen Softwareentwicklung und Gesellschaftsmodell’, Lehmanns Media, Berlin, S. 449–461. http://www.opensourcejahrbuch.de/2005/ [02. Feb 2006].

11 http://www.redhat.com/en_us/USA/home/company/news/prarchive/2005/press_3q_2006.html
12 Siehe Fußnoten 7 ff.