

# Hauptseminararbeit

Sommersemester 2005

Thema: Opensource-Geschäftsmodelle

Author: Michael Krauß

Fachsemester: 10

Studiengang: Wirtschaftsinformatik

Betreut durch: Prof. Dr. Ing. Hans Röck

Abgegeben am: 21.Juni 2005

Vorgetragen am: 10.Juni 2005

# Opensource-Geschäftsmodelle

6. Juli 2005

## Zusammenfassung

Diese Seminararbeit beschreibt Geschäftsmodelle, die sich um Opensource-Anwendungen in den letzten Jahren entwickelt haben. Nach der Erklärung der verwendeten Begriffe, stellt sie anhand der Wertschöpfungskette von Porter die unterschiedlichen Geschäftsmodelle vor und zeigt, dass Unternehmen im Opensource-Umfeld Geld verdienen. Die Arbeit endet mit einem Fazit und blickt auf Entwicklungen im Opensource-Umfeld.

## Inhaltsverzeichnis

1	Einleitung	1
2	Begriffe	1
2.1	Geschäftsmodell	1
2.1.1	Wertschöpfungskette von Michael Porter	2
2.2	Opensource	2
2.2.1	Besonderheiten von Opensource-Geschäftsmodellen	3
2.2.2	Gefahren für die Opensource-Bewegung	3
3	Geschäftsmodelle	4
4	Entwicklung	4
4.1	Applikationsentwickler	4
4.1.1	primäre Aktivitäten	4
4.1.2	unterstützende Aktivitäten	4
4.1.3	Abgrenzungsstrategien	5
4.2	Distributor	5
4.2.1	primäre Aktivitäten	5
4.2.2	Unterstützende Aktivitäten	6
4.2.3	Abgrenzungsstrategien	7
5	Dienstleistungen	7
5.1	Application Service Providing	7
5.1.1	primäre Aktivitäten	7
5.1.2	unterstützende Aktivitäten	7
5.1.3	Abgrenzungsstrategien	8
5.2	Beratungsunternehmen	8
5.2.1	primäre Aktivitäten	8
5.2.2	unterstützende Aktivitäten	9

5.2.3	Abgrenzungsstrategien	9
5.3	weitere Geschäftsmodelle	10
5.3.1	Bücher und Zeitschriften	10
5.3.2	Mediator	10
6	Hardware	10
6.1	Hardwareproduzenten	10
6.1.1	primäre Aktivitäten	10
6.1.2	unterstützende Aktivitäten	11
6.1.3	Abgrenzungsstrategien	11
7	Fazit und Ausblick	11
8	Glossar	12

## 1 Einleitung

Die klassische Ökonomie geht davon aus, dass sich der Preis eines Gutes über verfügbare und nachgefragte Menge bildet. Liefert kein Anbieter die nachgefragte Menge, steigt, wie momentan beim Erdöl zu beobachten, der Preis. Wieviel darf dann ein Gut kosten, das unbeschränkt zur Verfügung steht? — Nichts?!

Obwohl es sich bei Software, nachdem die Entwicklungskosten verdient wurden, um ein unbeschränkt zur Verfügung stehendes Gut handelt, verschenken Softwareunternehmen, wie Microsoft, SAP und Adobe Systems selten Programme<sup>1</sup> und veröffentlichen fast nie den dazugehörigen Quelltext. Statt dessen verknappten technische<sup>2</sup> und rechtliche<sup>3</sup> Maßnahmen künstlich das Angebot und sichern den Softwareunternehmen ihre Marktmacht und Preise.

Die Unternehmenskonzentration im Softwaremarkt hat in den letzten Jahren zugenommen und die Abhängigkeit der Nutzer von wenigen Softwareanbietern verstärkt. Halten Anbieter ihre Datenformate geheim<sup>4</sup> oder erweitern Standards durch proprietäre,

<sup>1</sup>Mit den verschenkten Programmen kann der Anwender Dokumente konsumieren, aber nicht produzieren.[9]

<sup>2</sup>z.B. Produktaktivierung bei Microsoft Windows XP

<sup>3</sup>Raubkopierer werden seit dem 13.09.2003 mit bis zu fünf Jahren Gefängnis bestraft

<sup>4</sup>Apple untersagt RealNetworks, die Verwendung des iTunes DRM (digital Rights Management) Fairplay und bietet keine Lizenzierung an. Kunden des iPod müssen bei iTunes DRM geschützte Musik kaufen, um sie auf ihrem iPod abzuspielen.[11]

nicht freigegebene Erweiterungen, werden Anwender zu Abhängigen, die dem Hersteller ausgeliefert sind. Der in Geld bewertete Verlust von Daten, die nicht mehr verarbeitet werden können, wenn der Wechsel auf ein anderes Textverarbeitungssystem erfolgt, übersteigt die gesparten Lizenzgebühren. Der Anwender bleibt beim Hersteller hängen. Ist das Erpressung?

Während es bis zum Anfang der 90er Jahre keine Alternative zu diesem Dilemma gab, beginnt nun die Uhr der freien Software zu schlagen. Am Anfang belächelt und unterschätzt, breitet sich der Freiheitsgedanke, der freier Software zu Grunde liegt, auf andere Medien wie Musik, Bilder, Texte und Wissen aus. Die Menge des frei zugänglichen Materials wächst seit Jahren.

Doch was passiert mit jenen, die freie Inhalte publizieren? Funktionieren ihre Geschäftsmodelle? Oder handelt es sich bei freier Software schreiben und Geld verdienen um ein Oxymoron? Diese Seminararbeit erklärt die verwendeten Begriffe und stellt anhand der Wertschöpfungskette von Porter Geschäftsmodelle vor, die sich durchgesetzt haben. Diese Arbeit ignoriert sowohl Geschäftsmodelle, die Opensource-Anwendungen intern einsetzen<sup>5</sup>, als auch solche, die versuchen Opensource-Anwendungen mit proprietären Anwendungen so zu bündeln, dass der Anwender von der proprietären Anwendung abhängt. Die ersten ignoriere ich, weil sie in Bereichen arbeiten, die dem Anliegen der Opensource-Bewegung nur indirekt zu Gute kommen, die zweiten, weil sie den Grundgedanken der Opensource-Bewegung nicht verstehen und den Begriff Opensource für ihr Marketing missbrauchen.

## 2 Begriffe

### 2.1 Geschäftsmodell

Die Literatur definiert den Begriff Geschäftsmodell mehrdeutig. Der Detaillierungsgrad eines Geschäftsmodells reicht dabei von einer kurzen Beschreibung, welchen Mehrwert ein Unternehmen schafft, bis zur komplexen Darstellung aller am Geschäftsmodell beteiligten Akteure und Prozesse. [4] vergleicht 28 Geschäftsmodelldefinitionen. Gemeinsam haben alle Definitionen, dass sie ein Geschäftsmodell direkt mit einem Unternehmen verbinden. Diese Arbeit betrachtet das Geschäftsmodell größer und fasst ähnliche Geschäftsmodelle zusammen, um die Aufgabenbereiche, die sich um Opensource-Anwendungen entwickelt haben, leichter zu gliedern. Der Ansatz von Michael Porter,

die Wertschöpfungskette, zeigt sich als geeignete Möglichkeiten ein Geschäftsmodell darzustellen, weil sich an ihr die Aktivitäten eines Geschäftsmodells ablesen lassen.

#### 2.1.1 Wertschöpfungskette von Michael Porter

Michael Porter[3] schreibt, dass der Mehrwert, den Unternehmen schaffen, sich als Summe aller Mehrwerte der einzelnen Prozesse ergibt. Die Kombination verschiedener Prozesse liefert ein Produkt oder eine Dienstleistung, die am Markt angeboten wird. Porter unterscheidet zwischen primären Aktivitäten, als jenen die die Kernkompetenzen des Unternehmens bilden und den sie unterstützenden Aktivitäten, die der Betrieb des Unternehmens verlangt, die aber keine Kernkompetenzen darstellen.

Als Michael Porter die Wertschöpfungskette entwickelte, gab es keine freie Software, weshalb er in seiner Arbeit der Frage: Was bedeutet Mehrwert? nicht nachging. Unternehmen, die alle erbrachten Leistungen vollständig am Markt verkaufen, stellen sich diese Frage nicht, weil sie versuchen für jede erbrachte Leistung den Mehrwert über den Marktpreis zu erzielen. Wenn Unternehmen Leistungen, z.B. Software, verschenken, stellt sich die Frage, ob diese Aktivitäten zu Wertschöpfungskette gehören oder nicht. In dieser Arbeit gehe ich davon aus, dass auch unbezahlte Aktivitäten zur Wertschöpfungskette gehören, da sie den realen Wert, definiert als tatsächlicher Nutzwert eines Gutes, für einen Nutzer, erhöhen.

Im Gegensatz zu [12], gliederte diese Arbeit die Aktivitäten, die zu einem Geschäftsmodell gehören, in primäre und sekundäre Aktivitäten. Diese Gliederung ist notwendig, weil Unternehmen aus den hier vorgestellten allgemeinen Geschäftsmodellen, ihre individuelle Wertschöpfungskette zusammenstellen müssen und so eine Wichtung der Aktivitäten erfolgt.

### 2.2 Opensource

Wer sich mit der Opensource-Bewegung auseinandersetzt, trifft auf die beiden Begriffe Opensource und Freie (free) Software. Beide Definitionen sehen in der Verfügbarkeit des Quelltextes die Voraussetzung, damit Software als frei oder offen bezeichnet werden kann. Während die Verfechter der freien Software bemängeln, dass die Opensource-Definition der Open Software Initiative nicht weit genug geht, weil sie den Freiheitsaspekt ignoriert[6], suchte die Open Source Initiative aufgrund der Doppeldeutigkeit des Wortes *free* im Englischen eine praktische und weniger ethische Definition von nicht proprietärer Software.

<sup>5</sup>Google setzte im Jahre 2000, ungefähr 5000 Linux Server ein[7]

Richard Stallman[5] führte den Begriff Free Software als Gegenpol zu proprietärer, geschlossener<sup>6</sup> Software 1982 ein. Das frei in Freie Software bezieht sich auf die Freiheit, die diese Software dem Nutzer garantiert. Also Freiheit, statt Freibier. Neben dem GNU<sup>7</sup>-Manifest, in dem Stallman philosophisch darlegt, warum Software frei sein müsse, schafft er mit der GNU Public License (GPL)<sup>8</sup> eine Lizenz, die es proprietären Softwareherstellern untersagt, freie Software in ihren proprietären Produkten einzusetzen. Stallman nennt vier Voraussetzungen, die Software erfüllen muss, um ihr das Attribut *frei* voranstellen zu dürfen.

- Die Freiheit ein Programm für jeden Zweck zu verwenden.

Sie erlaubt es jedem Anwender, ein freies Programm in jeder Einrichtung, für jeden Zweck und ohne Erlaubnis des Entwicklers zu verwenden.

- Die Freiheit, zu verstehen, wie ein Programm funktioniert und wie man es an eigene Bedürfnisse anpasst

Jeder Anwender darf Änderungen am Programm vornehmen, muss sie aber nicht veröffentlichen. Diese Freiheit schützt den Anwender davor, dass er vom Hersteller des Programmes gezwungen werden kann, Änderungen nur durch den Hersteller durchführen zu lassen. Wenn der Anwender seine Änderungen geheimhalten möchte, dann darf er das veränderte Produkt weder veröffentlichen noch verkaufen.

- Die Freiheit, ein Programm kostenlos zu verteilen

Die Nutzer dürfen Programme untereinander tauschen und an andere Nutzer kostenlos weitergeben.

- Die Freiheit ein Programm zu verändern und zu veröffentlichen

Jeder Nutzer darf das Programm ändern und seine Änderungen veröffentlichen.

Da sich die Definition [15] der Opensource Initiative von Freier Software nur durch die geringere Bedeutung der Freiheit unterscheidet, werden in dieser Arbeit die Begriffe Opensource und Freie Software synonym gebraucht. Wobei die Bedeutung, im Zweifel, der Stallmans Definition folgt.

<sup>6</sup>geschlossen=Der Hersteller veröffentlicht keine Quelltexte und verbietet die Weitergabe

<sup>7</sup>GNU=Gnu is not Unix (rekursives Akronym)

<sup>8</sup>Erklärung zur GPL und weiteren freien Softwarelizenzen findet sich hier: <http://www.opensource.org/licenses/index.php>

### 2.2.1 Besonderheiten von Opensource-Geschäftsmodellen

**Keine Lizeneinnahmen** Softwareproduzenten, die ihre Quelltexte und Dateiformate verheimlichen, verlangen von Nutzern Entgelte für die Nutzung dieser Software. Die Höhe des verlangten Preises spiegelt weder die Programmqualität noch die Herstellkosten wieder, sondern ist Maßstab der Marktmacht und der in der Vergangenheit aufgebauten Nutzerabhängigkeiten<sup>9</sup>. Freie Software bietet diese Einnahmemöglichkeit nicht. Obwohl die GPL ausdrücklich erlaubt, Geld durch den Verkauf der Software zu verdienen[8], beschränkt sie die Höhe des Preises auf die Kosten, die für die Vielfältigung der Software entstehen.<sup>10</sup> Kauft ein Nutzer eine GPL-lizenzierte Software, darf er sie kostenlos weitergeben.

**Kein Schutz des Quelltextes** Da freie Software per Definition keinen Besitzer hat, fehlt derjenige, der die Nutzung der Software durch andere untersagen kann. Anwender und Wettbewerber können die Quelltexte ansehen, verändern und für eigene Programme verwenden. Wenn ein Softwareproduzent sich entscheidet, seine Software unter einer freien Lizenz herauszugeben, gibt er einen Teil der Kontrolle über den Quelltext auf und riskiert, dass ähnliche Programme auf Basis seines Quelltextes<sup>11</sup> entstehen, aber er erhält dafür neue Ideen, wie sich die Software weiterentwickeln kann.<sup>12</sup> Desweiteren ist eine leichtere Fehlerbehebung möglich, wenn Nutzer statt Fehlermeldungen Fehlerlösungen (Patches) einschicken.

**Abhängig von freien Entwicklern** Je nach Geschäftsmodell entsteht in freien Softwareprojekten eine starke Abhängigkeit von freien Softwareentwicklern, d.h. freie Entwickler werden benötigt, um die gewünschte Software zu produzieren. Da sie vom Softwareproduzenten selten bezahlt werden, müssen andere Motivationsmethoden eingesetzt werden. Die Abhängigkeit entsteht, weil freie Softwareentwickler nur kostenlos an Programmen mitarbeiten, die sie interessieren, oder die sie benutzen.<sup>13</sup> Um die Abhängigkeit von freien Entwicklern zu reduzieren kann ein Rückgriff auf große etablierte Softwarepro-

<sup>9</sup>Eine Preisdifferenz von 600 EUR zwischen Microsoft Office und Openoffice spiegelt nicht den Qualitätsunterschied wieder.

<sup>10</sup>z.B. Betrieb eines Webservers, Pressen einer CD

<sup>11</sup>OpenBSD, NetBSD, MacOS und FreeBSD sind Beispiele für verschiedene Betriebssysteme, die auf dem Originalquellcode von BSD aufsetzen.

<sup>12</sup>Die Freigabe des Quellcodes des Netscape Webbrowser führte zu den heutigen freien Webbrowsern mozilla und firefox.

<sup>13</sup>Dies ist u.a. eine Erklärung dafür, warum es wenige freie Finanzbuchhaltungsprogramme gibt.

jekte erfolgen, wo das Ausscheiden eines Entwicklers eine kompensierbare Lücke hinterlässt.

### 2.2.2 Gefahren für die Opensource-Bewegung

Über die Chancen, die Opensource-Anwendungen bietet, findet sich im Internet genügend Material<sup>14</sup>, deshalb verzichte ich an dieser Stelle auf die Darstellung und stelle stattdessen die juristischen Entwicklungen vor, die die Weiterverbreitung von freier Software gefährden.

**Softwarepatente** Am 18. Mai 2005 beschloss der europäische Rat, Softwarepatente in der Europäischen Union einzuführen<sup>15</sup>. Softwarepatente gefährden die Entwicklung freier Software, weil viele *Standards*<sup>16</sup> durch Patente, insbesondere *Trivialpatente*<sup>17</sup>, geschützt sind.

Sobald Patentverwertungsgesellschaften von großen Softwareunternehmen beginnen, ihre Softwarepatente gegen freie Software durchzusetzen, werden freie Projekte sterben, denn kein freier Entwickler riskiert ein Rechtsstreit mit einem Konzern.

**Lizenzrisiko** Während Softwareunternehmen die Einhaltung ihrer Lizenzen streng überwachen und deren Missachtung hart bestrafen, nehmen sie die Lizenzbedingungen freier Software weniger ernst<sup>18</sup>. Leider hat sich die GPL als wichtigste Opensource-Lizenz, erst vor wenigen Gerichten durchgesetzt, Unternehmen die unter der GPL entwickeln stehen immer in der Gefahr, dass Wettbewerber ihre Innovation stehlen.

**Haftungsrisiko** In der Bundesrepublik Deutschland kommt zum Lizenzrisiko ein zusätzliches Haftungsrisiko für Programmfehler trotz GPL auf die Entwickler zu. Während es in den USA möglich ist, die Haftung für ein Produkt vollständig auszuschließen, ist dies in Deutschland unmöglich<sup>19</sup>.

**weitere Gefahren** Das deutsche Urheberrecht verbietet Kopierschutzmaßnahmen zu umgehen und geheimgehaltene Protokolle und Dateiformate per Reengineering zu erforschen und die Ergebnisse zu

<sup>14</sup>z.B. [16], [17]

<sup>15</sup>Die Entscheidung wurde unter fragwürdigen demokratischen Mitteln gefällt. Weitere Informationen: <http://www.softwarepatente.com/>

<sup>16</sup>z.B. Fortschrittsbalken, Hyperlinks

<sup>17</sup>die bisher nur in den Vereinigten Staaten und Japan durchsetzbar sind.

<sup>18</sup>Medion<sup>13</sup> nimmt die Lizenzbedingungen der GPL nicht ernst und veröffentlicht den Quelltext des geänderten Programms nicht

veröffentlichen. Es entsteht die Gefahr, dass Anwender beim Wechsel zu Opensource-Anwendungen ihre Daten verlieren, wenn der Formatbesitzer die Öffnung der Formate untersagt.

## 3 Geschäftsmodelle

Im folgenden Abschnitt werden die einzelnen Geschäftsmodelle vorgestellt. Tabelle 1 zeigt, dass sich Opensource-Geschäftsmodelle um Opensource-Anwendungen bilden. Ein Geschäftsmodell für Opensource kombiniert die um den Kern gruppierten Bereiche Hardware, Service, Support und Dokumentation und baut aus ihnen ein für den Anwender nutzbares Produkt. Der Verdienst mit reinen Opensource-Anwendungen, also dem Schreiben derselbigen, ohne unterstützende Tätigkeit, ist kommerziell unmöglich, weil jedes Unternehmen wenigstens Geld für die Gehälter der angestellten Entwickler benötigt. Im Gegensatz zu herkömmlichen IT-Geschäftsmodellen, *kann* man bei Opensource-Anwendungen Anpassungen und Verbesserungen selbst vornehmen, man *muss*, es aber nicht.

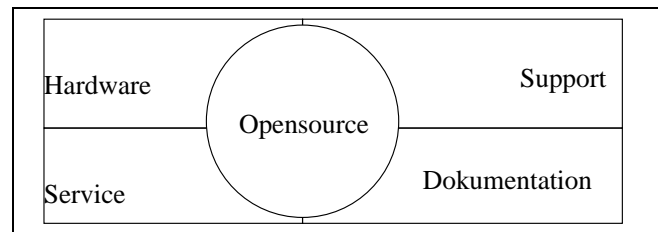


Abbildung 1: Opensource-Umfeld

## 4 Entwicklung

### 4.1 Applikationsentwickler

[12] klassifiziert die Applikationsentwickler nach den Lizenzen, die sie für ihre Produkte anbieten. Die Entwickler, die ihre Produkte nur mit freien Lizenzen anbieten und den Einsatz in proprietären Lösungen untersagen; und jenen, die zusätzlich eine proprietäre kostenpflichtige Lizenz anbieten, aber dafür dem Kunden erlauben den Quelltext seines Programms nicht veröffentlichen zu müssen<sup>19</sup>. Abbildung 2 zeigt die Wertschöpfungskette.

#### 4.1.1 primäre Aktivitäten

<sup>19</sup>Trolltech lizenziert die Bibliothek QT nach diesem Prinzip. <http://www.trolltech.com>

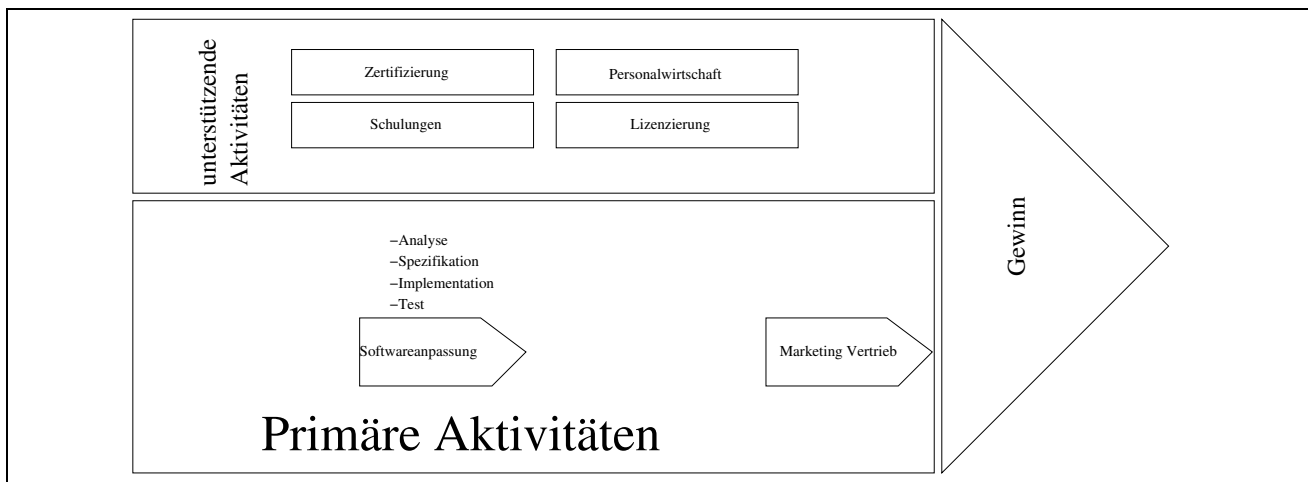


Abbildung 2: Wertschöpfungskette OS-Applikationsentwickler

**Softwareentwicklung** Die Softwareentwicklung umfasst alle Phasen, die bei der Erstellung von Software nach dem Wasserfallmodell durchlaufen werden, also Anforderungsanalyse, Systementwurf, Implementation und Test. Das ursprüngliche Produkt wird frei entwickelt und kostenlos abgegeben, deshalb konzentrieren sich die Entwicklungsaktivitäten auf die Anpassungen an Kundenwünsche, die Unterstützung anderer Programme oder die Portierung auf andere Betriebssysteme.

**Dokumentation** Abhängig vom Fokus des entwickelten Programms geht die Dokumentation besonders auf die Informationen ein, die Nutzer brauchen. Handelt es sich um eine Bibliothek oder geht die Dokumentation besonders auf die Entwickler ein. Wird die Dokumentation gedruckt und als Buch veröffentlicht, entsteht eine weitere Einnahmequelle für den Applikationsentwickler.

#### 4.1.2 unterstützende Aktivitäten

**Schulungen** Schulungen machen den Anwender/ Entwickler mit der produzierten Software vertraut. Die angebotenen Schulungen beschleunigen den Lernprozess beim späteren Anwender. Zu Schulungen zählt des weiteren die Organisation von Konferenzen auf denen Lösungen, vorgestellt werden, die mit der entwickelten Software erzielt wurden.

**Zertifizierung** Die Zertifizierung kann für Produkte, oder Menschen erfolgen. Werden Produkte zertifiziert, dann signalisiert man, dass die zertifizierten Produkte mit der entwickelten Software kompatibel sind oder sie benutzen. Erhalten Menschen Zertifikate, so weisen sie mit diesem Zertifikat nach, dass sie bestimmte Fähigkeiten der Software nutzen

können. Diese Zertifizierung ist für Unternehmen interessant, die sicherstellen müssen, dass Entwickler in einer Entwicklungsgruppe einen einheitlichen Wissensstand besitzen.

**Lizensierung** Die Lizenzierung gehört zur Wertschöpfungskette des Entwicklers, wenn er verschiedene Lizenzen, also proprietäre und freie anbietet.

**Zusammenarbeit mit anderen Projekten** Für Applikationsentwickler ist es wichtig, dass sie über eine breite Anwenderbasis verfügen. Besonders Applikationsentwickler, die Bibliotheken und keine Anwendungen herstellen, versuchen ihre Bibliothek zum Bestandteil von Projekten zu machen, die darauf aufbauen. Denn umso häufiger die eigene Bibliothek in fremden Projekten Verwendung findet, desto häufiger werden Schulungen, Zertifizierungen und Lizenzen nachgefragt.

#### 4.1.3 Abgrenzungsstrategien

Die im Opensource-Umfeld tätigen kommerziellen Applikationsentwickler haben sich auf die Entwicklung von Bibliotheken oder Basistechnologien konzentriert, weil diese Spezialisierung die Möglichkeit bietet, dass Kunden sich ihre Wünsche vom Applikationsentwickler herstellen lassen wollen und dafür bezahlen.

Dass Applikationsentwickler am Markt mit offenen Lösungen bestehen, zeigen die folgenden Beispiele: Trolltech (QT-Widgetbibliothek), Zend (Scriptsprache php) und OpenCMS( Contentmanagementsystem: OpenCMS). Neben diesen gibt es weitere Opensource-Projekte, die Anwendungen entwickeln; da es sich bei ihnen eher um Stiftungen handelt und

sie keine kommerziellen Interessen verfolgen, werden sie hier nicht aufgeführt.

## 4.2 Distributor

Es gibt heutzutage keine für einen Endanwender nutzbare Software, die nicht auf anderen Programmen oder Bibliotheken aufsetzt und deren Funktionalitäten erweitert. Gerade in freien Softwareprojekten ist es üblich, sich anderer freier Projekte zu bedienen. Da die Abhängigkeiten mit der steigender Komplexität der Software wachsen, ist es für den Anwender umständlich, die benötigten Abhängigkeiten aktuell zu halten. Täglich die neuesten Programmupdates suchen und installieren zu müssen, verhindert, abgesehen vom Aufwand, dass auf einem stabilen System gearbeitet wird.

1977 gibt die University of California die Berkeley Software Distribution heraus und eröffnet damit das Zeitalter der freien Softwaredistributionen. In regelmäßigen Abständen versorgte sie Anwender mit neuen Programmversionen und Sicherheitsupdates. Die Abbildung 3 zeigt die Wertschöpfungskette eines Distributors.

### 4.2.1 primäre Aktivitäten

**Forschung und Entwicklung** Unter Forschung und Entwicklung fasse ich die Aktivitäten zusammen, die die Basis einer Distribution legen und die nicht für jeden Distributionsrelease neu geschrieben werden. Da eine Distribution auf einem leeren System installiert wird, überlegt sich der Distributor, wie der Installationsprozess abläuft. Eine einfach zu benutzende Installationsroutine schafft Wettbewerbsvorteile. Programme entwickeln sich weiter, deshalb braucht die Distribution ein System für die Verwaltung der verschiedenen Programmversionen. Hier kann der Distributor eine existierende Paketverwaltung einsetzen oder eine eigene entwickeln. Vorteile der bewährten Paketverwaltungen: viele Projekte bieten bereits Pakete in diesem Format an, womit sich die Einbindung in die Distribution vereinfacht. Im Linuxmarkt haben sich bereits Paketverwaltungssysteme durchgesetzt, weshalb es hier keinen Sinn macht, ein neues System zu entwickeln. Nachdem der Anwender die Distribution installiert hat, braucht er Werkzeuge um die Programme zu konfigurieren. Standardmäßig befinden sich alle Konfigurationsdateien im Unixumfeld im */etc*, aber händische Konfiguration von Programmen, die miteinander arbeiten, ist aufwändig und fehleranfällig.

**Operationen** Unter Operationen subsumiere ich alle Tätigkeiten, die für die Herausgabe eines neuen

Releases erforderlich sind. Es werden die Aktionen in der Opensource-Szene beobachtet und Programme ausgewählt, die in die Distribution passen. Jede Distribution spezialisiert sich auf Anwendungsgruppen. Eine Serverdistribution enthält keinen graphischen Webbrowser und eine Distribution, die freie Programme verteilen möchte, schließt Programme, die unter keiner freien Lizenz stehen aus. Weiterhin müssen in diesem Prozess neue Versionen der Programme, getestet werden, die bereits zur Distribution gehören. Laufen sie stabil genug, werden sie Bestandteil des neuen Releases. Nachdem die Programme und Programmversionen für einen Release ausgewählt wurden, werden aus ihnen Pakete für den Release gebaut. Die Arbeit für einen neuen Release vereinfacht sich, wenn es dem Distributor gelingt, sein Paketformat bei Entwicklern beliebt zu machen, so dass diese eigene Programme im Paketformat abliefern können. Alternativ kann die Pflege einer Distribution auch dezentral durch eine Community erfolgen.

**Vertrieb** Distributionen können direkt, über Zeitschriften, und mit Computern gebündelt verkauft werden. Zunehmend setzen sich Breitbandanschlüsse durch, was die Anwender in die Lage versetzt, sich ihre Distribution selbst aus dem Internet zu laden. Dies führt dazu, dass Distributoren ihre Dienstleistungen erweitern, um neue Einnahmequellen zu erschließen, weil die Verkaufserlöse für die Distribution ausfallen.

### 4.2.2 Unterstützende Aktivitäten

**Bekannte Entwickler einstellen** Gelingt es einem Distributor bekannte Opensource-Entwickler für die eigene Distribution zu gewinnen, so wird die Distribution in der Opensource-Szene besser wahrgenommen. Angestellte Entwickler bringen Knowhow, das in die Distribution einfließt; die Qualität verbessert sich.

**Dokumentation** Dies ist nicht die richtige Stelle, die Bedeutung von Dokumentation für die Benutzbarkeit eines Programms hervorzuheben, aber eine Distribution, hat auch die Aufgabe, falls es für ein geliefertes Programm keine Dokumentation gibt, wenigstens eine Minimaldokumentation zu erstellen. OpenBSD hebt sich an dieser Stelle stark von anderen Distributionen ab, weil sie eine der detailliertesten und aktuellsten Dokumentation zu allen der Distribution angehörenden Paketen liefert.

**Schulung und Training** Damit Anwender die Distribution benutzen können, müssen sie geschult werden. Distributionsanbieter schulen Anwender, in

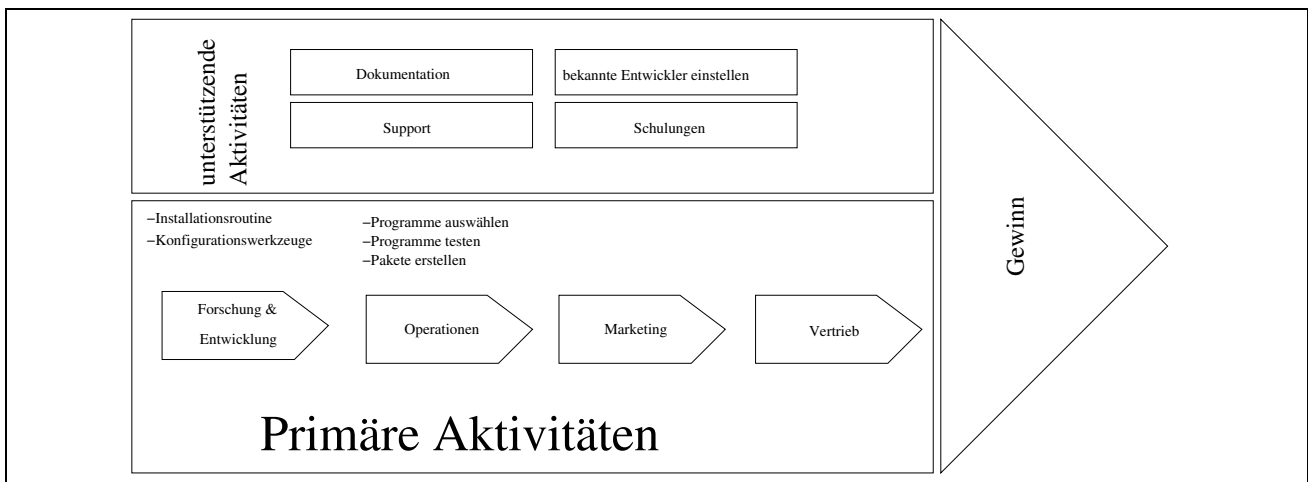


Abbildung 3: Wertschöpfungskette OS-Distributor

dem sie die Funktionen der Distribution vorstellen. Da Distributionen im Unixumfeld entstanden und auch heute noch weitestgehend dort zuhause sind, vermittelt die Schulung neben distributionsspezifischem Wissen, auch genügend Grundlagen, dass Anwender mit anderen Distributionen arbeiten können.

**Support** Insbesondere Unternehmenskunden erwarten von Distributoren einen kostenpflichtigen Support. Der Kunde schließt einen Supportvertrag mit dem Distributor statt vieler individueller Supportverträge mit den Entwicklern. Der Supportvertrag hat aus Sicht des Distributors den Vorteil, dass er kontinuierlich Geld einnimmt.

#### 4.2.3 Abgrenzungsstrategien

Wesentliche Abgrenzungsmöglichkeiten bestehen im Markt, auf dem der Distributor arbeitet. Spezialisiert er sich auf Geschäfts- oder Privatkunden? Eine weitere Abgrenzung erfolgt durch die Wahl des eingesetzten Betriebssystems und der Abwägung zwischen Stabilität von Programmen und deren Aktualität.

Laut Leiteritz in [12] reagieren Linuxdistributionskunden preiselastisch und da die Wechselkosten nicht durch die Höhe des Distributionspreises, sondern durch die Anpassung der bestehenden Installation und Konfigurationen entstehen, entwickeln in der Vergangenheit Distributoren proprietäre Konfigurations- und Installationsprogramme<sup>20</sup>, die im Gegensatz zum verteilten Betriebssystem, gerade nicht frei waren. Diese Strategie widersprach dem Gedanken freier Software und hätte, wäre sie fortgesetzt worden mittelfristig zu eine Situation geführt, wie sie auf dem Unixmarkt entstand, nachdem AT&T seine

Unixentwicklung einstellt hatte. Sun, HP und andere hatten das Ur-Unix um proprietäre Komponenten erweitert, die dazu führten, dass die neu entstandenen Systeme sich aufwendiger zusammensetzen ließen. Diese Gefahr haben die Linuxdistributoren erkannt und standardisieren deshalb die Linuxdistributionsen.<sup>21</sup>

Die auf dem Linuxmarkt aktiven Unternehmen RedHat, Suse und Mandrake expandieren in den Markt der Softwareberatung, der Supportverträge und Schulungen, um weitere Einnahmemöglichkeiten neben den Distributionseinnahmen zu generieren. Sie definieren sich als Systemhäuser, die dem Kunden eine vollwertige IT-Dienstleistung erbringen.

## 5 Dienstleistungen

### 5.1 Application Service Providing

Application Service Provider (ASP) verkaufen keine Software, sondern die Dienstleistung einen definierten Service nutzen zu können. Obwohl OpenSource-Anwendungen nichts kosten, verbrauchen Hardwareeinrichtung, Systemadministration und Softwarekonfiguration Ressourcen des Anwenders und kosten damit Geld. Wird der OpenSource-Entwickler zum ASP, so kann er seine Programme offen entwickeln, kostenlos verteilen und Geld verdienen. Da der Application Service Provider den Service anbietet, eine Offene Software nutzen zu können, kann der Kunde den Provider wechseln, sobald er mit dem empfangenen Service unzufrieden ist. Die Abhängigkeit die zwischen Kunden und OpenSource-ASP entsteht, ist kleiner, als die zwischen Kunde und proprietärem ASP, weil ein OS-ASP keinen anderen ASP von der Nut-

<sup>20</sup>z.B. Yust Another System Tool (YAST) der SuSe GmbH

<sup>21</sup>www.linuxbase.org

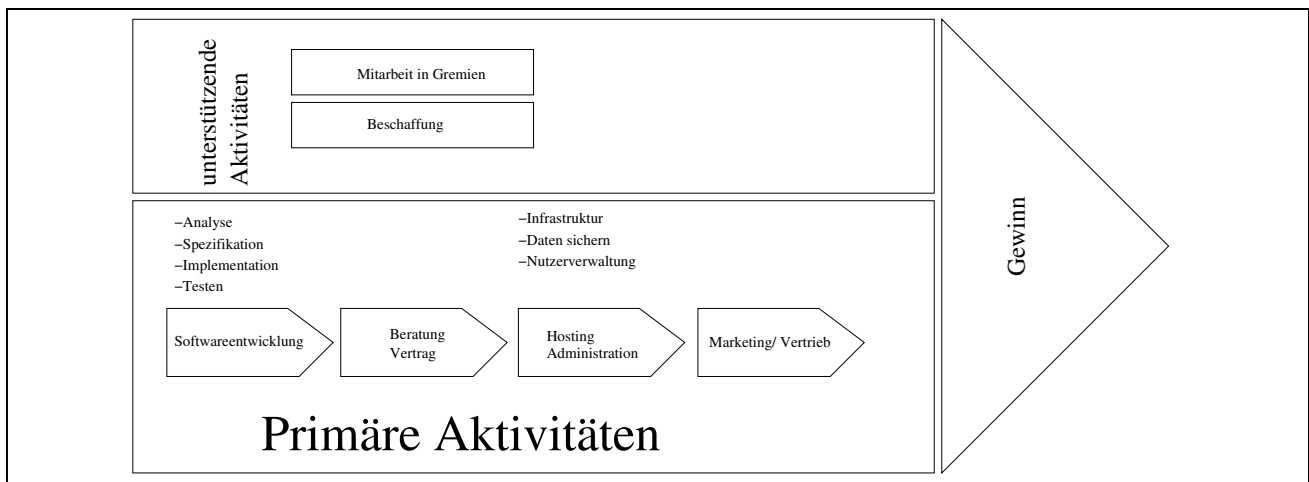


Abbildung 4: Wertschöpfungskette OS-Application Service Provider

zung seiner OS-Applikation ausschließen kann. Wenn er Opensource-Anwendungen entwickelt, stehen sie im Internet jedermann frei zur Verfügung, also auch Wettbewerbern.

### 5.1.1 primäre Aktivitäten

**Softwareentwicklung** Die Softwareentwicklung zählt zu einer primären Aufgabe eines ASP, wenn er eine selbst geschriebene Applikation anbietet. Die Softwareentwicklung beinhaltet alle Phasen des Lebenszyklusses eines Softwareproduktes. Die Softwareentwicklung muss nicht die vollständige Anwendung schaffen, sondern kann auch eine bestehende Opensource-Anwendung um Funktionen oder Module erweitern.

**Beraten und Vertragsgestaltung** Durch die Beratung erfährt der Provider, welche Anforderungen der Kunde an den Service stellt und welche Anpassungen der Standardsoftware notwendig sind. Am Ende der Beratung steht der Vertrag, der regelt, welchen Service der Provider erbringt und wieviel der Service kostet.

**Hosting** Der ASP hostet die Applikation, d.h. er stellt die Infrastruktur bereit, auf der die Applikation und die von ihr benötigten Services laufen sollen. Hosting bedeutet auch, dass er für die Hardwarekosten aufkommt und fehlerhafte Hardware ersetzt.

**Anwendungsadministration** Die Anwendungsadministration beinhaltet Aufgaben, die sicherstellen, dass die angebotene Applikation immer auf dem neuesten Stand ist. Dazu zählen das Einspielen von Updates, das kontinuierliche Testen der Anwendung, die Sicherung der Anwenderdaten sowie die Verwaltung der Applikationsnutzer.

### 5.1.2 unterstützende Aktivitäten

**Mitarbeit in Gremien und externen Projekten** Die Mitarbeit in Gremien oder anderen Opensource-Projekten ermöglicht dem ASP seinen Kunden das verfügbare Knowhow zu zeigen, denn die Mitarbeit in einem Gremium setzt Fachwissen voraus. Ein weiterer Positiver Effekt durch die Mitarbeit in Gremien ist, dass eigene Positionen und Entwicklung leichter als Standard vereinbart werden können.

**User Support** Der Application Service Provider betreut den Nutzer und löst dessen Probleme. Er steht als Ansprechpartner zur Verfügung, wenn der Nutzer Probleme mit Software hat.

### 5.1.3 Abgrenzungsstrategien

Die heutigen ASP die Opensource-Anwendungen einsetzen, haben sich auf Webapplikationen konzentriert. So bieten OS-Contentmanagementsystem-Hersteller (CMS) an, das CMS bei ihnen zu hosten. Andere ASP hosten die offenen Systeme, ohne selbst Applikation zu entwickeln.

Wie Beratungsunternehmen spezialisieren sich ASP auf Branchen und deren Bedürfnisse. Neben der Branchenspezialisierung, kann ein ASP sein Angebot zusätzlich auf Unternehmen mit einer bestimmten Betriebsgröße (KMU<sup>22</sup>, Konzerne) zu schneiden.

## 5.2 Beratungsunternehmen

2002 fanden 28% aller Beratungen im IT-Sektor statt[2] Unternehmen hängen im zunehmenden Maße von einer funktionierenden IT-Infrastruktur ab, da aber

<sup>22</sup>Kleine und Mittlere Unternehmen

die eingesetzten Programme zunehmend komplexer werden, bietet sich eine externe Beratung an, weil der Aufbau einer IT-Infrastruktur selten zu den Kernkompetenzen eines Unternehmens gehört. Das Beratermodell ist kein neues Geschäftsmodell, was durch Opensource-Bewegung entstand, sondern lediglich durch die Konzentration auf Opensource-Anwendungen eine höhere Glaubwürdigkeit beim Kunden erreicht. Die höhere Glaubwürdigkeit resultiert aus der Unabhängigkeit der Beratung, weil der Berater kein proprietäres Programm empfiehlt, sondern freie Lösungen. Wenn das Programm nichts kostet, verdient der Berater nichts am Verkauf, er steht in keinem Interessenkonflikt, sondern kann sich auf den technischen Vergleich konzentrieren. Die Wertschöpfungskette Abbildung 5 basiert auf der Wertschöpfungskette von Dienstleistungsunternehmen, die auf Projektbasis arbeiten und die [1] vorstellt.

Leider ist es zur Zeit noch nicht möglich, alle betrieblichen Anwendungen durch gleichwertige Opensource-Anwendungen zu ersetzen, aber vielleicht gelingt dies, wenn Beratungsunternehmen die Nachfrage bündeln und Projekte starten können.

### 5.2.1 primäre Aktivitäten

**Akquise** Die Akquise gewinnt neue Beratungsaufträge für das Beratungsunternehmen.

**Kontaktphase** Die Kontaktphase beschreibt die Phase, in der, nachdem ein Beratungsauftrag gewonnen wurde, die Beratungstätigkeit stattfindet. In einem ersten Schritt erfolgt die Analyse der bisherigen IT-Infrastruktur (Hardware und Software) und das Beratungsunternehmen verschafft sich einen Überblick über die Aufgaben, die der Kunde mit seiner IT-Infrastruktur lösen möchte. Der Berater sucht Opensource-Anwendungen, die die geforderten Aufgaben lösen bzw. welche Anpassungen entweder an den Programmen oder Anforderungen notwendig sind. In dieser Phase prüft er wie die Daten aus bisherigen Formaten in die neuen Formate konvertiert werden können. Weiterhin wird überprüft, ob die Hardware ausreicht. Nachdem die Programme ausgewählt wurden, schlägt er einen Deploymentleitfaden vor, d.h. einen Zeitplan, wie die neuen Programme installiert werden können und in welchen Schritten vorgegangen wird. So werden zuerst die Dienste auf Opensource-Anwendungen umgestellt, die für den Nutzer transparent sind, z.B. Verzeichnisdienste, Webserver und Datenbankserver. Erst in einem zweiten Schritt erfolgt die Installation von Programmen auf den Desktops der Mitarbeiter, auch hier wird wenn möglich zweistufig vor-

gegangen. Zunächst bleiben alte und neue Programme parallel installiert und erst im zweiten Schritt erfolgt die endgültige Entfernung des ursprünglichen Programms[14]. Mitarbeiter sammeln in dieser Phase erste Erfahrungen mit der neuen Software und werden gleichzeitig geschult werden.

**Nachkontaktphase** Nachdem die Beratung abgeschlossen und umgesetzt wurde, bleibt der Berater im Kontakt mit dem Beratenen. Er überprüft, ob sich die erwarteten Ergebnisse eingestellt haben und behebt wenn nötig die auftretenden Probleme. Die Nachkontaktphase ist wichtig, weil der Berater so erfährt, wie sich Programme in der Praxis bewähren. Er wird zur Schnittstelle zwischen Entwicklern und Kunden.

### 5.2.2 unterstützende Aktivitäten

**Personalwirtschaft** Die Mitarbeiter sind Wettbewerbsvor- und -nachteil zugleich. Die Auswahl von Mitarbeitern, die in Opensource-Projekte involviert sind, sichert dem Beratungsunternehmen, dass die Berater wissen, was Programme können und was nicht. Desweiteren braucht ein Beratungsunternehmen Mitarbeiter, die einen Überblick über die Opensource-Entwicklungstrends haben und den global sich im IT Markt abzeichnenden Trend erkennen.

**Technologieentwicklung** Ein Beratungsunternehmen kennt die Opensource-Entwicklungstrends und versucht durch eigenes Engagement diese mit zu gestalten. Weitere Aufgaben der Technologieabteilungen besteht darin, Programme auf ihre Eignung für den produktiven Einsatz zu untersuchen und die Opensource-Anwendungs Spreu vom Weizen zu trennen.

**Beschaffung** Es geht um die Beschaffung der Komponenten, die das Beratungsunternehmen benötigt, um die Programme zu testen und eventuell eigene Programme zu entwickeln.

**Dokumentation** Neben der für Beratern obligatorischen Veröffentlichung von Erfolgsgeschichten<sup>23</sup>, kann durch eine genaue Dokumentation der durchgeführten Tätigkeiten weiteres Vertrauen in die Arbeitsweise des Beraters aufgebaut werden. Potentielle Kunden sehen, wie das Beratungsunternehmen an konkreten Beispielen vorgegangen ist.

<sup>23</sup>Positive Fallstudien, die zeigen, wie ein Beratungsunternehmen ein Kundenproblem löst.

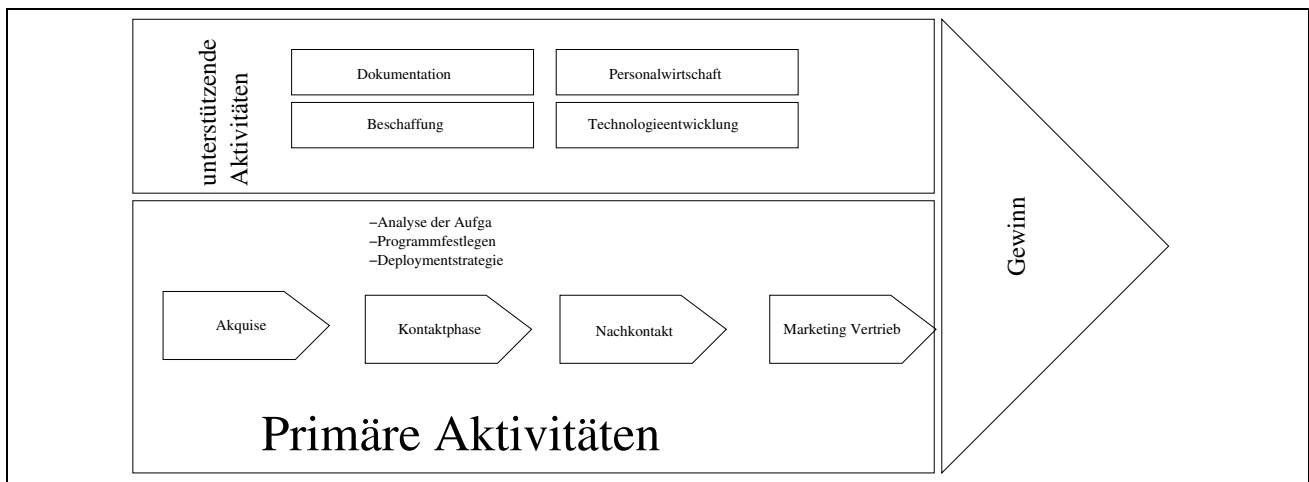


Abbildung 5: Wertschöpfungskette OS-Beratungsunternehmen

### 5.2.3 Abgrenzungsstrategien

Unterschiedliche Branchen stellen unterschiedliche Anforderungen an ihre IT Infrastruktur. So braucht z.B. das Südstadtklinikum andere Anwendungen, als die Kundenbetreuung der WIRO. Spezialisiert sich ein Beratungsunternehmen auf eine Branche, kann es sich dort positionieren.

Ähnlich des in der Opensource-Szene verbreiteten Community Gedanken, also dem gemeinsamen Entwickeln einer Anwendung, kann ein Beratungsunternehmen versuchen, die Interessen seiner Kunden zu bündeln und Vorteile in der gemeinsamen Entwicklung oder der Kommunikation mit Entwicklern zu erreichen.

Eine weitere Abgrenzung bietet sich einem Beratungsunternehmen, wenn es sich entscheidet, wieviele der Informationen, die bei einer Beratung entstanden sind, freizugeben. Wenn das Beratungsunternehmen viele Informationen veröffentlicht erleichtert es anderen Opensource-Beratungsunternehmen ähnliche Projekte in Gebieten durchzuführen, die das Beratungsunternehmen nicht abdecken kann.

Der Beratungsmarkt mit reinen Opensource-Beratungsunternehmen entwickelt sich und große OS-Beratungsunternehmen existieren nicht. Momentan versuchen Beratungsunternehmen eine Brücke zwischen Opensource-Anwendungen und proprietären Anwendungen zu schlagen, indem sie *das Beste* aus beiden Welten anbieten. Der Autor sieht diese Bestrebungen kritisch, wenn Beratungsunternehmen nicht mittelfristig das Ziel verfolgen, die proprietäre Anwendung vollständig durch eine offene zu ersetzen.

## 5.3 weitere Geschäftsmodelle

Es gibt neben den bisher vorgestellten Opensource-Geschäftsmodellen weitere Geschäftsmodelle, die ich kurz anreißen möchte.

### 5.3.1 Bücher und Zeitschriften

Oft liest der Anwender lieber ein Buch, um eine neue Technologie oder Anwendung zu verstehen, als 500 Seiten zu drucken oder sie am Bildschirm zu lesen. Deshalb gibt es Verlage, die Opensource-Literatur (Howtos, Referenzen, Handbücher, etc) verlegen. Diese Verlage entwickeln weder Opensource-Anwendungen noch schreiben sie die publizierten Texte, sondern sie helfen wie jeder Verlag seinen Autoren, ein Buch zu produzieren, das der Leser versteht. Es gibt Opensource-Entwickler, die Bücher schreiben und mit den Einnahmen ihren Lebensunterhalt bestreiten<sup>24</sup>. Verfolgt der Verlag eine liberale Handhabung seines Copyrights, so stellt er die vollständigen Bücher unter einer offenen Lizenz zum Download bereit.

Zeitschriften verdienen, wie Buchverlage Geld, indem sie ihre Leser über aktuelle Entwicklungen in der Opensource-Szene informieren und neue Opensource-Anwendungen und Opensource-Technologien vorstellen.

### 5.3.2 Mediator

Unter einem Mediator versteht [12] das Bindeglied zwischen Entwicklern und Interessenten, die eine IT Dienstleistung benötigen. Der Mediator baut eine Plattform auf, die die Profile von Entwicklern oder

<sup>24</sup>Larry Wall, Erfinder der Skriptsprache Perl, gibt das Perllehrbuch heraus.

Unternehmen veröffentlicht. Braucht ein Kunde eine bestimmte Dienstleistung, so vermittelt der Mediator den Kontakt zwischen Nachfrager und Anbieter. Der Mediator erhält eine Prämie, die ihm ein Entwickler für einen vermittelten Auftrag bezahlt. Bei diesem Geschäftsmodell existiert ein sich selbst verstärkender Sog, der alle Marktteilnehmer zum größten Mediator zieht. Dieser Sog (Netzwerkeffekt), entsteht, weil die Wahrscheinlichkeit den *richtigen* Auftraggeber bzw. Auftragnehmer, mit der Anzahl der bereits bei diesem Mediator registrierten Entwickler zu finden steigt. Hat ein Mediator eine solche Position erreicht, z.B. sourceforge.net<sup>25</sup>, fällt es anderen Anbietern sehr schwer, ähnlich erfolgreich zu sein, so setzt sich Berlios.de<sup>26</sup> als alternativer Mediator aus diesem Grund nicht durch. [12] nennt als weitere Einnahmequelle, den Verkauf von Büchern und Werbebannern.

## 6 Hardware

### 6.1 Hardwareproduzenten

Das Geschäftsmodell des Hardwareproduzenten habe ich in diese Arbeit aufgenommen, weil die Opensource-Szene dem Hersteller die Möglichkeit bietet Entwicklungskosten für den Hersteller zu sparen. Unterstützen Hardwareproduzenten Opensource-Betriebssysteme, so versprechen sie sich eine größere Nachfrage nach den eigenen Produkten und eine einfachere Nutzung der Hardware in Opensource-Betriebssystemen [18]. Es handelt sich um eine Win Win Situation: Der Hersteller verringert sein Engagement in der Treiberentwicklung; der Anwender kann das Gerät unter verschiedenen Betriebssystemen betreiben. Auch im Embedded Devices Markt gibt es Hersteller, die die Vorteile von Opensource-Betriebssystemen aufgreifen und Geräte produzieren, auf denen ein freies Betriebssystem läuft. Der Vorteil ist auch hier: Die Konzentration auf die Entwicklung der Hardware, und kleinere Anpassungen an das Betriebssystem. Diese Anpassungen sind erforderlich, damit das ursprüngliche Betriebssystem mit der eigenen entwickelten Hardware arbeitet.

#### 6.1.1 primäre Aktivitäten

**Forschung und Entwicklung** Wie bei allen Hardwareproduzenten existiert eine Forschungs- und Entwicklungsabteilung, die neue Hardware entwirft. Der einzige Unterschied zu Hardwareproduzenten, die ihre Geräte nicht auf offene Betriebssysteme abstimmen,

<sup>25</sup>Ein Anbieter, der Opensourceanwendungen hostet

<sup>26</sup>bietet ähnliche Funktionen wie sourceforge und wurde staatlich gefördert

besteht darin, dass die hier vorgestellten Produzenten stärker auf Standards achten und weniger proprietäre Erweiterungen enthalten.

**Produktion** Die Produktion erfolgt durch den Hardwareproduzenten. Auch wenn er keine Fabrik baut, legt er fest, welche Geräte, wann und wie oft gebaut werden.

**Treibererstellung** Die Erstellung der Treiber ist neben der Produktion der Faktor, der über Markterfolg und -misserfolg eines Gerätes entscheidet. Wenn ein Hersteller auf Opensource-Betriebssysteme setzt, kann er bereits beim Schreiben des Gerätetreibers versuchen, eine Opensource-Gemeinschaft aufzubauen, bzw. eine bereits bestehende Entwicklergemeinde mit der Entwicklung des Treibers beauftragen. Schafft es der Hersteller eine Gemeinde zu finden, die den Treiber für verschiedene Betriebssysteme pflegt, braucht er sich um die Weiterentwicklung praktisch nicht mehr kümmern und zieht sich auf die Moderation der Entwicklung zurück.

**Marketing** Aufgabe des Marketing ist bei diesem Geschäftsmodell, die besondere Bedeutung und den Vorteil, den ein offener Gerätetreiber dem Anwender bietet, hervorzuheben.

#### 6.1.2 unterstützende Aktivitäten

**Dokumentation** Einen Treiber durch ein Opensource-Projekt (weiter)entwickeln zu lassen setzt voraus, dass die Hardware und die Funktionen dokumentiert sind. Umso besser die Dokumentation, desto einfacher können Gerätetreiber geschrieben werden und umso einfacher können Wettbewerber die selbst entwickelten Geräte nachbauen. Aber da der Originalhersteller bereits eine laufende Produktion hat und über Lernkurveneffekte geringe Stückkosten hat, sollte sich der Erstentwickler durchsetzen. Linux hat gezeigt, dass Computernutzer mehr Geld für Hardware bezahlen, wenn sie wissen, dass es von Linux unterstützt wird. Dies garantieren aber nur die Originalgeräte.

**Support** Der Support macht die gemeinsam mit der Opensource-Gemeinschaft entwickelten Treiber für den Nicht-Informatiker benutzbar und hilft, bei Fragen der Nutzer.

#### 6.1.3 Abgrenzungsstrategien

Der Hardwaremarkt ist riesig und die Diversifikationsmöglichkeiten für Unternehmen auch. Die verschiedenen Möglichkeiten zu erläutern, würde den

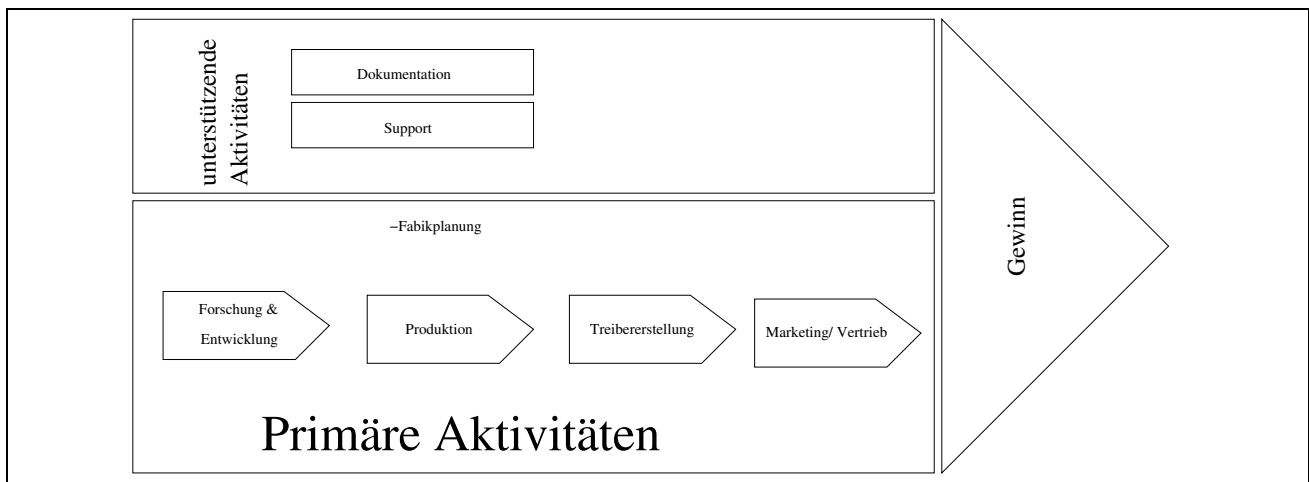


Abbildung 6: Wertschöpfungskette OS-unterstützender Hardwareproduzenten

Rahmen dieser Arbeit sprengen. In Bezug auf für Opensource besonders geeignete Hardware zu entwickeln, sehe ich die Möglichkeit entweder darin, neue technische Standards zeitnah in konkrete Produkte umzusetzen (WLAN), oder erwachsene Technologien (z.B. Netzwerkkarten) in möglichst hoher Stückzahl günstig herzustellen.

Der Hersteller von Servern haben die Bedeutung von Linux erkannt und investieren hohe Geldbeträge in die Weiterentwicklung des Betriebssystems Linux.

## 7 Fazit und Ausblick

Die Arbeit hat gezeigt, dass die Lizenzeinnahmen nicht Bestandteil eines Geschäftsmodells sein müssen, sondern dass sich Opensource-Geschäftsmodelle als Serviceangebote um Opensource-Anwendungen bilden. Die wichtigsten Unterschiede zwischen dem herkömmlichen IT-Geschäftsmodellen und Opensource-Geschäftsmodellen sind, dass Kunden zunehmend nur für genutzte Dienstleistungen bezahlen und dass Kunden unabhängiger von ihrem Softwareanbieter werden. Die gezeigten Opensource-Geschäftsmodelle arbeiten profitabel, wenn Kunden Änderungen einarbeiten lassen, Beratungen oder Service in Anspruch nehmen. Diese Geschäftsmodelle sind aus diesem Grund fairer, als auf proprietäre Software aufsetzende Modelle, weil sie dem Anwender für gleiches Geld mehr als das Nutzungsrecht an der Software einräumen.

Die Opensource-Bewegung führt dazu, dass die Interoperabilität von Programme zunimmt. Offene Formate erlauben, dass Programme Daten fremder Programme leichter importieren können. Wettbewerbsvorteile die durch den Einsatz von proprietären und geheimen Formaten entstehen, können nicht mehr entstehen. Unternehmen, die an Opensource-Trend

profitieren wollen, müssen sich andere Vorteile aufbauen.

Noch steht die Opensource-Bewegung am Anfang, weshalb sich zukünftig weitere Geschäftsmodelle entwickeln werden. Durch Migration von Unternehmen und Verwaltungen weg von proprietären System hin zu offenen, werden Anwender gezwungen sich mit Offenen System stärker auseinanderzusetzen<sup>27</sup>.

## 8 Glossar

### Opensource-Community (Opensource-Gemeinschaft)

Eine Gruppe von Menschen, die gemeinsam an einem Opensource-Projekt arbeitet.

**Lizenz** Die Lizenz ist ein Vertrag zwischen dem Anbieter einer Software und dem Nutzer einer Software. Sie regelt, was der Anbieter dem Nutzer für Rechte einräumt, was er ihm untersagt und welche Kosten er durch die Verwendung der Software zu tragen hat.

**Paket** Ein Paket enthält ein Programm, die dazugehörigen Bibliotheken, Konfigurationsdateien, Dokumentationen und Informationen über die Abhängigkeit von anderen Paketen. Pakete des selben Programms unterscheiden sich durch die Versionsnummern des Programms.

<sup>27</sup>Die Linuxmigration in München und die Schulung der Mitarbeiter wird die Hemschwelle sich Linux auf dem privaten Rechner zu installieren senken und zu einer weiteren Verbreitung führen.

---

## Literatur

### Literatur

- [1] Martin Benkenstein, *Strategisches Marketing*, W. Kohlhammer Drucker GmbH, Stuttgart, 2000, ISBN: 3-17-017001-5
- [2] *Consulting Market in Europe 2002*  
<http://www.feaco.org/images/downloads/Anlagen/Anlage%20Market%20Overview.pdf> 2005-04-13
- [3] Michael Porter, *Wettbewerbsvorteile (Competitive Advantage)*, Campus Verlag, Frankfurt, 2000, ISBN: 0743260872
- [4] *Geschäftsmodelle und internetbasierte Geschäftsmodelle – Begriffsbestimmung und Teilnehmermodell*  
<http://isym.bwl.uni-mainz.de/publikationen/isym012.pdf> 2005-04-13
- [5] Richard M. Stallman, *Definition of Free Software*  
<http://www.gnu.org/philosophy/free-sw.html> 2005-04-13
- [6] *What is Free Software? What is Open Source?*  
<http://www.free-soft.org/> 2005-04-13
- [7] *Interview with Google's Sergey Brin*  
<http://linuxgazette.tolix.org/issue59/correa.html> 2005-06-05
- [8] *GNU General Public License*  
<http://www.gnu.org/copyleft/gpl.html> 2005-06-05
- [9] Volker Grassmuck, *Freie Software - zwischen Privat und Gemeineigentum*
- [10] Horst Speichert, *Gewährleistungs- und Haftungsausschluss*  
<http://www.medienkultur-stuttgart.de/thema02/2archiv/news6/mks6OSS.htm> 2005-04-13
- [11] *iPod-Streit: Apple wirft RealNetworks "Hacker-Taktik" vor*  
<http://www.heise.de/newsticker/meldung/49586> 2005-06-05
- [12] Raphael Leiteritz, *Open Source-Geschäftsmodelle*  
[http://ig.cs.tu-berlin.de/osjb/content\\_OekonomieOSSGeschaeftsmodelle.htm](http://ig.cs.tu-berlin.de/osjb/content_OekonomieOSSGeschaeftsmodelle.htm) 2005-06-05
- [13] *Medion notebooks sold at ALDI violate GNU GPL*  
<http://www.gpl-violations.org/news/20050430-medion-aldi-notebook.html> 2005-06-05
- [14] *Migrationsleitfaden*  
[http://www.kbst.bund.de/download/mlf\\_v1\\_de.pdf](http://www.kbst.bund.de/download/mlf_v1_de.pdf) 2005-06-05
- [15] *The Open Source Definition Version 1.9*  
<http://www.opensource.org/docs/definition.php> 2005-06-05
- [16] *Warum soll man Open Source einsetzen?*  
[http://kb5.at/content/e22/e1022/index\\_ger.html](http://kb5.at/content/e22/e1022/index_ger.html) 2005-06-20
- [17] *Open Source Software*  
<http://oss-broschuere.berlios.de/broschuere/broschuere-de.html> 2005-06-20
- [18] *XGI setzt auf Open-Source-Treiber*  
<http://www.heise.de/newsticker/meldung/58503> 2005-06-05