

Software - eine Gefahr für die Demokratie?

Problemanalyse und Lösungsstrategie

Matthias Kirschner

9. Januar 2006



Wirtschafts- und Sozialwissenschaftliche Fakultät

Wintersemester 2005/2006

Dr. Viktoria Kaina

Is democracy working?

Herausforderungen und Leistungsfähigkeit eines politischen Ordnungsmodells

Inhaltsverzeichnis

1	Überblick	1
2	Verschiedene Arten von Software	1
3	Problemanalyse	3
3.1	Unfreie Software schwächt die Wirtschaft	3
3.2	Gefährliche Machtverteilung	5
4	Strategische Schritte	6
4.1	Bildung	6
4.2	Eigene IT Projekte mit Freier Software umsetzen	7
4.3	Bewusster Gebrauch von BGMs	8

1 Überblick

Dieses Papier beschäftigt sich mit der Frage, welche Gefahren für die demokratische Gesellschaft von Software ausgehen. Für eine funktionierende Demokratie sind politische Gleichheit sowie Partizipation der Bürger von großer Bedeutung. Politische Gleichheit und wirtschaftliche Gleichheit korrelieren positiv (vgl. Mead 2004). Ebenso hängen Partizipationsmöglichkeiten und -bereitschaft unter anderem vom wirtschaftlichen Status ab (vgl. Schultze 2004, S. 648).

Bei Software muss zwischen unfreier und Freier Software unterschieden werden. Unfreie Software stellt einen Eingriff in die Freiheiten der Menschen dar und führt zwangsläufig zu wirtschaftlicher Ungleichheit. Eine Lösung dieses Problems ist die *Verwendung Freier Software*, sowohl in der *Bildung* als auch in *eigenen IT Projekten* des Staates. Des Weiteren ist ein *sorgfältiger und gezielter Einsatz von begrenzten geistigen Monopolen (BGMs) in der Gesetzgebung* notwendig. Die Probleme und Empfehlungen sollen in den folgenden Kapiteln näher erläutert werden.

2 Verschiedene Arten von Software

Software kann sehr gut mit einem Kochrezept verglichen werden (vgl. Stallman 2001). Der Autor schreibt eine Liste von Anweisungen nieder aus deren Ausführung ein bestimmtes Ergebnis resultiert. Bei Computerprogrammen werden die Anweisungen im so genannten Quelltext niedergeschrieben. Es gibt eine Vielzahl von Programmiersprachen, die man dazu verwenden kann. Der Quelltext wird anschließend mit Hilfe eines Programmes, dem Compiler, in maschinenlesbare Form gebracht. Diese maschinenlesbare Form kann dann vom Computer ausgeführt werden. Sie ist jedoch von Menschen nicht mehr interpretierbar, da sie nur aus Nullen und Einsen besteht. In Abbildung 1 sieht man den Quelltext und den schematisierten Maschinencode eines Programmes, welches in der Programmiersprache C geschrieben ist. Wird es kompiliert und ausgeführt, gibt es „Hallo Welt!“ auf dem Bildschirm aus.

Bei Software muss zwischen zwei Modellen unterschieden werden: der Freien Software und der unfreien Software. Freie Software gewährt dem Nutzer folgende *vier Freiheiten*: (1) *unbegrenzte Nutzung zu jedem Zweck*, (2) *Studium und Anpassung*, (3) *Weitergabe durch Kopie* und (4) *Weiterentwicklung* (vgl. Stallman 2002, S. 41ff, Stallman 1996) . Unfreie Software gewährt keine, oder nicht alle

<code>#include <stdio.h></code>	0100101000100010010010
	1010010100010101001001
	1010001000001000011100
<code>int main(void)</code>	1000100100100100100010
<code>{</code>	1000010010010001000100
<code>printf("Hallo Welt!\n");</code>	0000100101000001000100
<code>return 0;</code>	1000100010001000001000
<code>}</code>	1000100101000010111101
	1000100100010000100001

Abbildung 1: Menschenlesbarer Quellcode und maschinenlesbarer Binärcode (eigene Darstellung)

dieser Freiheiten.

Die vier Freiheiten müssen in der Lizenz der Software gewährt werden. Innerhalb der Freien Software Lizenzen wird noch einmal zwischen stark schützenden, schwach schützenden und nicht schützenden Lizenzen unterschieden (vgl. Reiter 2004, S. 85-87):

Starker Schutz / Copyleft - z.B. GNU GPL: gewährt die vier Freiheiten und schützt sie dadurch, dass die Lizenz vererbt wird. Dies „impft“ die Software dagegen, wieder unfrei zu werden.

Schwacher Schutz / Copyleft - z.B. GNU LGPL: gewährt die vier Freiheiten. Jedoch bieten schwach schützende Lizenzen nur einen begrenzten Schutz der Freiheit. Im Gegensatz zu stark schützenden Lizenzen darf unfreie Software gegen Programme unter der GNU LGPL gelinkt werden. Das bedeutet, dass zwar das Programm selbst immer frei bleiben muss, es jedoch mit anderen Programmen kombiniert werden kann, welche unfrei sind.

Kein Schutz - z.B. XFree86 License (modifizierte BSD): gewährt ebenso die vier Freiheiten, bietet aber keinen Schutz der Freiheit. Das bedeutet, dass das Programm oder Programmteile auch wieder unfrei gemacht werden dürfen. Dies hätte zur Folge, dass Benutzern die Freiheiten vorenthalten werden.

Die Abbildung 2 veranschaulicht die verschiedenen Softwarekategorien (siehe Reiter 2004, Free Software Foundation 1996). Wichtig hierbei ist, dass der gezahlte Preis für den Erwerb der Software bei dieser Definition keine Rolle spielt.

Ausschlaggebend für die Unterscheidung zwischen Freier Software und unfreier Software ist nur, ob die vier Freiheiten gewährt werden oder nicht.

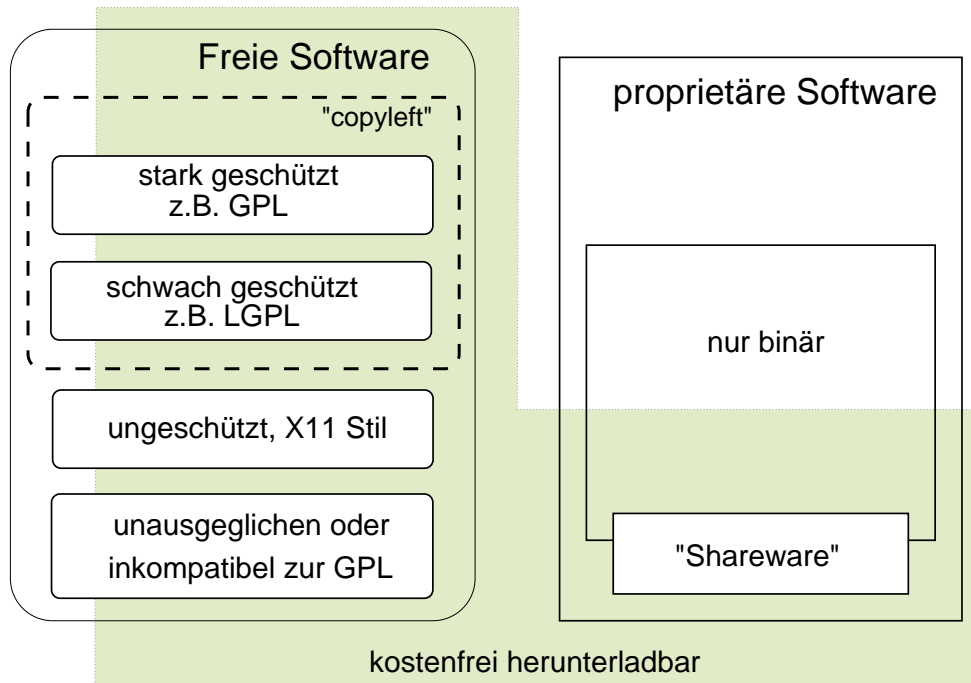


Abbildung 2: Software Kategorien nach (Reiter 2004, S. 86)

3 Problemanalyse

Software durchdringt heutzutage alle Bereiche unseres Lebens. Das Klingeln des Weckers, das Stellen der Stoppuhr für das Frühstücksei, das Öffnen der Straßenbahntüre, die Benutzung des Fahrstuhls, das Telefonieren mit dem Mobiltelefon und natürlich die klassische Variante, das Benutzen des Heimcomputers; immer kommen wir mit Software in Kontakt. Nahezu jedes elektronische Gerät enthält Software.

Wie stark wir von Software abhängig sind wird uns klar, wenn wir uns vorstellen welche Folgen es hätte, wenn die Geräte nicht funktionieren würden.

3.1 Unfreie Software schwächt die Wirtschaft

In unserem privaten Leben könnten wir mit den Auswirkungen einige Zeit zurechtkommen, auch wenn es uns nicht leicht fallen dürfte. Die Wirtschaft jedoch ist

auf funktionierende Software angewiesen. Eine Studie der Fraunhofer Gesellschaft hat ergeben, dass 50% der deutschen Industrie und 80% der Exporte von der Informations- und Kommunikationstechnologie abhängig sind (vgl. Miller 2004). Für sie ist es fatal, wenn die Software nicht funktioniert. Förderbänder stehen still, Flugzeuge und Züge verkehren nicht, die Börse würde zusammenbrechen und fast die gesamte Büroarbeit wäre unmöglich.

Software ist also ein zentraler Bestandteil unserer Wirtschaft. Das Problem dabei ist, dass unfreie Software immer zu einem Monopol führt. Warum dies so ist wollen wir nun etwas näher betrachten.

Für Geschäfte ist Kommunikation notwendig. Um Geschäfte tätigen zu können, müssen wir mit Kunden und Anbietern kommunizieren. Angebote müssen eingeholt, Verträge verschickt und Konzepte diskutiert werden. Ohne Kommunikation kann kein Geschäft stattfinden. Diese Kommunikation findet zu einem Großteil mit Hilfe von Software statt. Vermutlich werden in Zukunft immer mehr Geschäfte in elektronischer Form abgewickelt werden. Um Geschäfte ausführen zu können benötigen wir also Software.

Unfreie Software funktioniert nur mit sich selbst gut. Problematisch ist, dass unfreie Software meist nur mit sich selbst gut funktioniert. Jedem wird es schon einmal widerfahren sein: eine Datei, die z.B. mit einem bestimmten Textverarbeitungsprogramm geschrieben wurde, lässt sich nur mit dem gleichen Programm wieder fehlerfrei öffnen und betrachten. Oft ist es sogar nur mit der exakt gleichen Version der Textverarbeitung möglich.

Dies ist eine zwangsläufige Folge des unfreien Softwaremodells, denn das Geschäftsmodell basiert darauf, möglichst viele Lizenzen der Software zu verkaufen. Für diesen Zweck wird es einerseits dem Benutzer schwer gemacht, zu einer anderen Software zu wechseln, andererseits werden seine Kommunikationspartner dazu gezwungen, ebenfalls diese Software einzusetzen. Dies wird dadurch erreicht, dass das Programm es zunächst ermöglicht, standardisierte Dateiformate zu öffnen. Speichert man diese jedoch wieder ab, können die Daten nicht mehr mit anderen Programmen, die den Standard befolgen, geöffnet werden¹. Hierfür werden der Datei Erweiterungen hinzugefügt, die von anderen Programmen nicht unterstützt werden.

Daraus folgt: benutzen wir unfreie Software, müssen wir alle das gleiche Programm verwenden, um miteinander kommunizieren zu können. Das führt zu ei-

¹Diese Praktik ist recht verbreitet und wird als „Vendor Lock-In“ bezeichnet.

nem *Monopol beim Angebot von Software*². Für dieses Monopol müssen alle Bereiche der Wirtschaft bezahlen und die gesamte Volkswirtschaft wird durch das unfreie Softwaremodell geschwächt. Mit diesem Monopol halten einige wenige Unternehmen eine solche Marktmacht in ihren Händen, dass dadurch der Gesellschaft ein immenser Wohlfahrtsverlust entsteht. Die soziale Kluft wird dadurch immer größer.

3.2 Gefährliche Machtverteilung

„Alle Mittel, welche die Realisierung von Zwecken sozialer Akteure ermöglichen; die Akteure also mit Macht ausstatten“ werden als Machtressourcen bezeichnet (Weiß 2004, S. 499). Mit Software kann man sehr gut eigene Ziele durchsetzen. Der Benutzer kann die Regeln, welche in Software implementiert sind nicht missachten. Bei Software deren Quelltext nicht verfügbar ist, existiert nicht einmal die Möglichkeit, festzustellen, welche Regeln überhaupt implementiert sind. Eventuell werden manche dieser Regeln, wenn überhaupt, erst durch Zufall entdeckt.

American Online (AOL) bietet ein gutes Beispiel dafür, wie Computerprogramme Freiheiten einschränken (vgl. Lessig 1999, S. 66-71). Als Mitglied bei AOL hat man die Möglichkeit, mit anderen Mitgliedern synchron Nachrichten in einem virtuellen Diskussionsraum auszutauschen. Dazu muss man sich in diesen Diskussionsraum einloggen. Nun hat aber AOL in ihrer Software die Regel implementiert, dass nur 23 Personen pro Raum zugelassen sind. Eine Gruppe von 24 Personen hat also nicht die Möglichkeit, sich gemeinsam in dem virtuellen Diskussionsraum auszutauschen. Die Software macht hier keine Ausnahmen.

Zu diesen softwareimmanenten Beschränkungen kommen noch Beschränkungen durch das Urheberrecht hinzu. Wie machtlos dagegen sogar ganze Staaten sind, illustriert folgendes Beispiel:

„Um seine Schrift in der digitalen Welt zu bewahren und lebendig zu halten, bat [Island im Jahre 1998] Microsoft, eine Unterstützung für Isländisch in Windows zu implementieren. Es war sogar bereit, für diese Arbeit zu bezahlen, doch Microsoft sah den Markt als zu klein an und winkte ab. Ohne Zugang zum Quellcode und ohne das Recht, ihn zu modifizieren, ist das Land vollkommen abhängig von Microsofts Gnade“ (Grassmuck 2002, S. 318).

²Dieses Monopol bleibt nicht auf den Softwaremarkt begrenzt. Es dehnt sich auch auf den Hardwaremarkt aus. So läuft z.B. das Microsoft Betriebssystem nur auf Intel kompatibler Hardware und Computer mit Intel Chips werden mit dem Microsoft Betriebssystem vertrieben.

Eine Machtressource also vollkommen in den Händen einiger Weniger zu belassen, stellt eine große Gefahr für die demokratische Gesellschaft dar.

4 Strategische Schritte

Was können wir tun, um diese Gefahren zu verhindern? Welche Schritte können wir unternehmen, um unsere Wirtschaft vor Monopolen zu schützen? Wie können wir den Menschen ein Verständnis für eine wichtige Technik unserer Zeit vermitteln und dadurch verhindern, dass diese als Machtressource missbraucht wird?

Die einfachste Methode, die Gefahren von Software abzuwenden, wäre eine gesetzliche Bestimmung, dass jede Software für jeden Zweck verwendet, studiert und angepasst, durch Kopie weitergegeben und weiterentwickelt werden darf; also jede Software frei sein muss. Da jedoch meiner Ansicht nach solche Veränderungen Zeit benötigen, wollen wir hier mehrere kleinere Lösungsschritte betrachten.

4.1 Bildung

Es ist wichtig, dass die Fähigkeit Software zu verstehen nicht in den Händen einiger Weniger liegt, sondern breit in der Gesellschaft verteilt ist. Die Fähigkeit Programmieren zu können darf ebenso wenig wie Lesen, Schreiben oder Rechnen Herrschaftswissen sein. Die Bürger sollten in der Schule mit Rüstzeug ausgestattet werden, um ihre späteren Arbeitswerkzeuge kontrollieren zu können. Sie sollten also auch Grundkenntnisse im Programmieren erlernen, um später nicht den Irrglauben zu haben, Software sei etwas Übermenschliches.

Mit Freier Software können wir des Weiteren wichtige demokratische Prinzipien in der Bildung durchsetzen (vgl. Free Software Foundation Europe 2004, Stallman 2003):

Freiheit Die Schüler können mit Freier Software lernen, solche zu benutzen und zu verstehen. Dadurch, dass sie den Quelltext verfügbar haben, gibt es für ihren Wissensdurst keine Grenze. Niemand verbietet den Schülern nur bis zu einem bestimmten Grad die Software zu verstehen. Anders als bei unfreier Software können sie auf dem aktuellen Stand der Technik lernen und müssen sich nicht damit abfinden, dass manche Dinge geheim sind. Sie werden erkennen, dass es nicht nur immer einen Weg, oder ein Programm zur Lösung eines Problems gibt, sondern fast immer Alternativen bestehen.

Gleichheit Es besteht eine Gleichheit zwischen den Benutzern. Die Schule kann allen Schülern, auch denen aus ärmeren Familien, die Software zur Verfügung stellen. So sind arme Schüler nicht dazu gezwungen, gegen Gesetze zu verstoßen, damit sie nicht benachteiligt sind und die Software auch zu Hause benutzen können.

Brüderlichkeit Mit Freier Software lernen Schüler, dass sich Zusammenarbeit und gegenseitige Hilfe lohnen. Niemand vermittelt ihnen den Eindruck, dass es eine schlimme Tat ist, anderen zu helfen, indem Programme getauscht werden, und dass solche Praktiken nur von „Piraten“ verübt werden (vgl. Stallman 1994a).

4.2 Eigene IT Projekte mit Freier Software umsetzen

Zunächst sollte der Staat darauf achten, bei seinen eigenen IT Projekten selbst Freie Software Komponenten zu verwenden. Ansonsten könnte ihm ein Schicksal, wie das in Kapitel 3.2 beschriebene widerfahren.³

Des Weiteren wird durch die Verwendung von Freier Software sichergestellt, dass die Funktionsweise der Software überprüfbar ist. Und jeder Bürger sollte das Recht haben, zu wissen, welche Regeln der Staat implementiert hat, genauso wie er das Recht hat, erlassene Gesetze nachzulesen.

Die derzeitige Gesetzgebung verbietet es, das System zur elektronische Patientenkarte näher zu untersuchen. Dadurch wird es nahezu unmöglich gemacht zu überprüfen, ob dabei die Privatsphäre geschützt bleibt und in diesem Sinne der Sozialstaat aufrecht erhalten wird. Bernhard Reiter kritisiert, dass „[d]ie Systementwickler [...] den Datenmißbrauch wissentlich in Kauf [nehmen] und versuchen - mit Hilfe des Urheberrechts - eine Überprüfung des Sicherheitskonzepts zu verhindern. Hier wird die Gefahr deutlich, die Softwarepatente und die Verschärfung des Urheberrechts für die Gesellschaft darstellen“ (Free Software Foundation Europe 2005).

Wenn der Staat für alle IT Projekte Freie Software verwendet, kann diese auf legale Weise überprüft werden. So können Gefahren erkannt und frühzeitig darauf reagiert werden. Das Wissen und die Fähigkeiten von staatlich finanzierter Software sollte außerdem der Gesellschaft zur Verfügung stehen. So muss in den USA „Software, die mit staatlichen Mitteln entwickelt wurde, allen zugute kom-

³Die Isländische Regierung setzte die Lokalisierung schließlich auf der Basis des freien Betriebssystems GNU/Linux durch (vgl. Grassmuck 2002, S. 318).

men“(Grassmuck 2002, S. 381). Dieser Praxis sollte sich auch die Bundesrepublik anschließen. Staatlich geförderte Software sollte immer unter eine stark schützende oder zumindest schwach schützende Freien Softwarelizenz gestellt werden um sicherzustellen, dass diese auch frei bleibt. Sonst würden sich Unternehmen an der Gemeinschaft bereichern können, ohne ihr wieder etwas zurückzugeben (vgl. Grassmuck 2002, S. 306).

4.3 Bewusster Gebrauch von BGMs

Verhinderung von Monopolen im Softwarebereich kann am besten dadurch erreicht werden, dass die BGMs so gestaltet werden, dass sie ihren ursprünglichen Sinn auch verfolgen.

Unter „begrenzten geistigen Monopolen“ (BGM) verstehen wir Patente, Urheberrecht, Schutzmarken, sowie andere Gesetze, deren Absicht es ist, limitierte Monopole für geistige Kreativität zu gewähren (vgl. Greve 2003, S. 35).

Meist wird heutzutage der Eindruck vermittelt, „dass naturgegebene Rechte für Autoren die akzeptierte und unumstrittene Tradition unserer Gesellschaft sind“ (Stallman 1994b). Der Sinn und Zweck sowohl des angloamerikanischen Copyrights, als auch des europäischen Urheberrechts, war es jedoch, den Fortschritt zu fördern und nicht die Autoren zu belohnen. Sie belohnen zwar „die Autoren etwas und die Verleger etwas mehr, aber dies ist gewollt als ein Mittel, um ihr Verhalten zu verändern“ (Stallman 1994b). Genauso wurden Patente als „ein zeitlich begrenztes Monopol zur Förderung der Veröffentlichung und Verbreitung von wirtschaftlich interessanten Ideen“ geschaffen (Reiter 2004, S. 90).

Des Weiteren sollte bedacht werden, dass manche dieser Systeme, wie z.B. das Urheberrecht, schon sehr alt sind und auch für andere Technologien als Software geschaffen wurden. So entstand das Copyright-System

„mit der Drucktechnik - eine Technologie, die Kopien in Massenproduktion ermöglichte. Das Copyright passte gut zu dieser Technologie, da es nur die Massenproduzenten von Kopien einschränkte. Es nahm den Lesern von Büchern keine Freiheit. Eine gewöhnliche Leserin, die keine Druckerpresse besaß, konnte Bücher nur mit Stift und Tinte kopieren, und sehr wenige Leser wurden dafür verklagt.

Digitaltechnologie ist flexibler als die Druckerpresse: Wenn Information in digitaler Form vorliegt, kann man sie leicht kopieren, um sie mit

anderen zu teilen. Genau diese Flexibilität passt schlecht zu einem System wie dem Copyright“ (Stallman 1994b).

Das unfreie Softwaresystem wird, wie wir gesehen haben, immer zu einem Monopol führen. Es ist auf Dauer nicht machbar, immer nur die Folgen des Systems zu bekämpfen, ohne die Ursachen anzugehen. Wie schwierig sich diese Folgenbekämpfung in der Praxis darstellt, sehen wir bei dem Kartellrechtsverfahren der Europäischen Kommission gegen Microsoft. Hierbei lässt Microsoft nichts unversucht, um die Netzwerkschnittstellen von Microsoft Windows nicht offenlegen zu müssen. Durch dieses Verhalten verhindern sie jegliche Konkurrenz ⁴.

Der Gesetzgeber muss sich dessen bei seiner Aufgabenerfüllung immer bewusst sein. BGMs müssen mit Bedacht verwendet werden, um eine gesellschaftliche Entwicklung zu ermöglichen. Und speziell bei Software sollte sehr gewissenhaft darauf geachtet werden, ob das Ziel mit diesem Mittel erreicht wird oder ob es nicht doch bessere Möglichkeiten dafür gibt.

⁴Microsoft hat in dem Verfahren mittlerweile mehr als drei Milliarden US Dollar, das ist das Sechsfache der von der Europäischen Kommission auferlegten Strafe, an Drittparteien des Berufungsverfahren vor dem Europäischen Gerichtshof bezahlt. Die Drittparteien bekundeten daraufhin kein weiteres Interesse an dem Fall zu haben. In Folge dessen unterstützen derzeit nur noch zwei Parteien die Europäische Kommission in dem Verfahren.

Literatur

Free Software Foundation 1996

FREE SOFTWARE FOUNDATION: *Categories of Free and Non-Free Software*.
<http://www.gnu.org/philosophy/categories.html> vom 05.01.2006, 1996

Free Software Foundation Europe 2004

FREE SOFTWARE FOUNDATION EUROPE: *Warum Freier Software in Schulen den Vorzug geben?* <http://www.fsfeurope.org/projects/education/argumentation.de.html> vom 05.01.2006, 2004

Free Software Foundation Europe 2005

FREE SOFTWARE FOUNDATION EUROPE: *Diagnosen verboten: elektronische Patientenakte krankt an der Sicherheit*. <http://mail.fsfeurope.org/pipermail/press-release-de/2005q4/000081.html> vom 05.01.2006, 2005

Grassmuck 2002

GRASSMUCK, Volker: *Freie Software zwischen Privat- und Gemeineigentum*.
Bonn : Bundeszentrale für politische Bildung, 2002

Greve 2003

GREVE, Georg C. F.: Fighting Intellectual Property: Who Owns and Controls the Information Societies? In: HEINRICH BÖLL FOUNDATION (Hrsg.): *Visions in process, world summit on the information society*. Geneva 2003, Tunis 2005, 2003, S. 35–38

Lessig 1999

LESSIG, Lawrence: *Code And Other Laws Of Cyberspace*. New York, NY, 1999

Mead 2004

MEAD, Lawrence: The Great Passivity. In: *Perspectives on Politics* 2 (4) (2004), S. 671–675

Miller 2004

MILLER, Franz: Innovationstreiber Informations- und Kommunikationstechnik. 2004. – Forschungsbericht

Reiter 2004

REITER, Bernhard E.: Wandel der IT: Mehr als 20 Jahre Freie Software. In: *Praxis der Wirtschaftsinformatik* HMD 238 (2004), August, S. 83–91

Schultze 2004

SCHULTZE, Rainer-Olaf: Partizipation. In: NOHLEN, Rainer-Olaf (Hrsg.): *Lexikon der Politikwissenschaft* Bd. 2. München : C.H. Beck, 2004, S. 647–649

Stallman 1994a

STALLMAN, Richard M.: *The Right to Read*. <http://www.gnu.org/philosophy/right-to-read.html> vom 05.01.2006, 1994

Stallman 1994b

STALLMAN, Richard M.: *Warum Software keine Eigentümer haben sollte*. <http://www.gnu.org/philosophy/why-free.de.html> vom 05.01.2006, 1994

Stallman 1996

STALLMAN, Richard M.: *The Free Software Definition*. <http://www.gnu.org/philosophy/free-sw.html> vom 05.01.2006, 1996

Stallman 2001

STALLMAN, Richard M. ; POTTONEN, Hannu (Hrsg.): *Intro des Filmes The Code*. 2001. – Film.

Stallman 2002

STALLMAN, Richard M. ; GAY, Joshua (Hrsg.): *Free Software Free Society: Selected Essays of Richard M. Stallman*. 1. Boston : GNU Press, 2002

Stallman 2003

STALLMAN, Richard M.: *Why schools should use exclusively free software*. <http://www.gnu.org/philosophy/schools.html> vom 05.01.2006, 2003

Weiß 2004

WEISS, Ulrich: Machtressourcen. In: NOHLEN, Rainer-Olaf (Hrsg.): *Lexikon der Politikwissenschaft* Bd. 1. München : C.H. Beck, 2004, S. 499