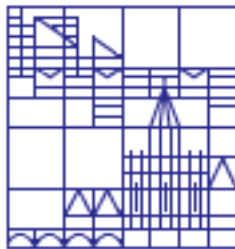


Kulturtechnik Software

Warum Organisationen, die mit Entwicklungsländern kommunizieren, Freie Software einsetzen müssen

Matthias Kirschner

7. November 2005



Universität Konstanz

Fachbereich Politik und Verwaltungswissenschaft

Kurs: Projekte und Personal in der Entwicklungszusammenarbeit (Gruppe1)

Dozent: Dr. Günther Oldenbruch

Zusammenfassung

Um eine nachhaltige Entwicklung zu ermöglichen, muss Software, als Kulturtechnik des 21. Jahrhunderts, den Menschen in den Entwicklungsländern zur Verfügung stehen. Hierfür müssen Organisationen selbst konsequent Freie Software einsetzen.

Inhaltsverzeichnis

1	Einleitung	2
2	Technische Grundlagen von Software	2
3	Freie Software - Frei wie in Freiheit	3
4	Bedeutung von Software	6
4.1	Politische Auswirkungen	7
4.2	Wirtschaftliche Auswirkung	9
5	Rolle der Organisationen	11
6	Fazit	12

1 Einleitung

Vom September 2004 bis April 2005, war ich im Rahmen meines Arbeitsaufenthalts, für die Free Software Foundation Europe (FSFE) tätig. Die FSFE dient Politikern, Rechtsanwälten, Journalisten und Softwareentwicklern als Ansprechpartner zum Thema „Freiheit in der Informationsgesellschaft“ und insbesondere für „Freie Software“. Im Zuge des Praktikums kam ich, unter anderem, mit dem UN World Summit on the Information Society (WSIS) in Berührung. Bei dem UN WSIS handelt es sich um den Weltgipfel zur Informationsgesellschaft der Vereinten Nationen, der im Dezember 2003 in Genf stattfand und dessen zweiter Gipfelpunkt im November 2005 in Tunis sein wird. Wie bei anderen Weltgipfeln geht es darum, die weltweiten Regeln und Visionen gemeinsam festzulegen, um diese dann national umzusetzen: In diesem Fall also die globalen Regeln der Informationsgesellschaft.

In dieser Zeit habe ich die Erfahrung gemacht, dass immer mehr Organisationen im WSIS Umfeld zwar Freie Software propagieren, jedoch selbst unfreie Software einsetzen. Dadurch wird es Entwicklungsländern nahezu unmöglich gemacht, Freie Software zu verwenden.

Lesen, Schreiben, Rechnen und Ackerbau haben, wie alle Kulturtechniken, einen entscheidenden Einfluss auf die wirtschaftlichen, politischen und gesellschaftlichen Perspektiven eines Landes. Software, als Kulturtechnik des 21. Jahrhunderts, muss allen Menschen gleichermaßen zur Verfügung stehen (vgl. Greve 2003). Deshalb tragen Organisationen, welche mit Entwicklungsländern kommunizieren, eine besondere Verantwortung bei ihrer eigenen Softwarewahl. Wenn sie selbst keine Freie Software einsetzen, schränken sie die Entwicklungsmöglichkeiten der Länder stark ein.

2 Technische Grundlagen von Software

Software kann sehr gut mit einem Kochrezept verglichen werden¹. Der Autor schreibt eine Liste von Anweisungen nieder und nach ihrer Ausführung steht

¹vgl. Richard M. Stallman in dem Film „The Code“ (2001), Regie und Buch: Hannu Pottonen. Making Movies und ADR Productions.

```
#include <stdio.h>
int main(void)
{
    printf("Hallo Welt!\n");
    return 0;
}
```

0100101000100010010010
1010010100010101001001
1010001000001000011100
1000100100100100100010
1000010010010001000100
0000100101000001000100
1000100010001000001000
1000100101000010111101
1000100100010000100001

Abbildung 1: Menschenlesbarer Quellcode und maschinenlesbarer Binärkode

ein bestimmtes Ergebnis. Bei Computerprogrammen werden die Anweisungen in dem 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.

3 Freie Software - Frei wie in Freiheit

Im September 1983 kündigte Richard M. Stallman sein Vorhaben an, ein völlig freies Betriebssystem zu schreiben; mit dem Namen GNU, für „GNU’s not Unix“. Er beendete seine Arbeit beim MIT und begann, Anfang 1984, das erste „bewusste“ Projekt mit Freier Software zu schreiben (vgl. Grassmuck 2002, S. 226). Mit Erfolg, spätestens seit dem Jahr 1994 steht das wohl bekannteste freie Betriebssystem, GNU/Linux, in einer stabilen Version zur Verfügung. Auch unter anderen Betriebssystemen dürfte Freie Software heutzutage bekannt sein. Etwa in Form des Webbrowsers Mozilla Firefox, des Compilers GCC oder des Videoplayers VLC. Spätestens wenn man das Internet be-

nutzt, kommt man zwangsläufig mit Freier Software in Berührung. Dabei ist sie jedoch für den Anwender oft unsichtbar. So sind z.B. der Webserver Apache und der Internet Domain Name Server BIND, die meist verwendeten Programme ihrer Art, ohne die das Internet nicht funktionieren würde.

Der Begriff, Freie Software² selbst, wurde zum ersten Mal vollständig im (GNU's Bulletin 1987) definiert. Danach muss Freie Software, um als solche zu gelten, folgende Freiheiten gewähren (vgl. Gay 2002, S.41 ff.):

Freiheit 1: Die unbegrenzte Nutzung zu jedem Zweck. Die Lizenz darf niemanden von der Benutzung der Software ausschließen. Eine Klausel, nach der die Software nicht in bestimmten Nationen verwendet werden darf, sowie jegliche andere Diskriminierung, macht die Software unfrei. Des Weiteren darf die Lizenz nicht verbieten, dass die Software für bestimmte Zwecke eingesetzt wird. Die Entscheidung, welche Aufgaben mit dem Programm gelöst werden sollen, liegt alleine in den Händen des Anwenders.

Freiheit 2: Studium und Anpassung an eigene Bedürfnisse.

Jedem soll es möglich sein, die Funktionsweise der Software, sofern er das will, zu erlernen. Sei es aus Gründen der Erkenntnisgewinnung oder um die Software seinen eigenen Bedürfnissen anzupassen. Anpassungen dürfen entweder selbst oder durch andere gemacht werden. Für diese Freiheit ist die Verfügbarkeit des Quelltextes zwingend notwendig. Um auf die Analogie des Kochrezeptes noch einmal einzugehen: Welchen Nutzen hätte ein Kochrezept, bei dem es dem Koch nicht erlaubt wäre, es abzuändern; z.B. Zutaten, die er nicht mag, wegzulassen, oder andere hinzuzufügen?

Diese Freiheiten sind dafür bestimmt, sich selbst zu helfen. Freie Software geht aber darüber hinaus. Sie soll es auch ermöglichen, anderen zu helfen, bzw. sich von anderen helfen zu lassen. Dies ist gerade in Anbetracht dessen, dass nicht jeder programmieren kann oder lernen will, notwendig.

²Freie Software begegnet einem manchmal auch unter dem Namen „Libre Software“ oder „Open Source Software“. Es wird jedoch empfohlen den Begriff „Freie Software“ zu verwenden. Dieser ist, im Gegensatz zu den anderen, klar definiert (vgl. Reiter 2004, S. 84-85).

Freiheit 3: Weitergabe durch Kopie. Jeder hat das Recht das Programm an andere, sei es gegen Entgelt oder gratis, weiterzugeben. Dadurch kann man anderen helfen, ihre Aufgaben mit dem Programm zu lösen, auch wenn dieses ursprünglich nicht dafür gedacht war. Dies steigert die Verbreitung der Software und maximiert den Gesamtnutzen der Gesellschaft, da die zur Verfügung stehenden Ressourcen, hier Programmierer, optimal genutzt werden.

Freiheit 4: Weitergabe von Modifikationen. Weiterhin muss die Freiheit gegeben sein, auch Änderungen wieder veröffentlichen zu dürfen. Man muss Änderungen nicht weitergeben, wenn man das nicht will, man hat aber das Recht dazu dies zu tun.

Damit eine Softwarelizenz als freie Softwarelizenz gilt, muss der Autor, in der Lizenz der Software, diese vier Freiheiten gewähren. Ist dies nicht der Fall, wird die Software unfrei (proprietär) genannt. Es existieren sehr viele verschiedene Freie Software Lizenzen, wobei die am weitest verbreitetsten die GNU General Public License (GNU GPL), die GNU Lesser General Public License (GNU LGPL) sowie X11-artige Lizenzen sind. Auf eine Darstellung der verschiedenen Lizenzmodelle Freier Software soll hier aber verzichtet werden. Eine gute Beschreibung dieser findet sich bei (Reiter 2004, S. 85-87).

Abbildung 2 veranschaulicht die verschiedenen Softwarekategorien. An dieser Stelle soll noch einmal hervorgehoben werden, dass die Bezeichnung „Freie Software“ unabhängig vom für den Erwerb des Programms gezahlten Preis ist. Die Bezeichnung bezieht sich ausschließlich auf die vier Freiheiten. Freie Software darf verkauft werden und nach einer Studie von (Lakhani et al. 2002, S. 38), werden über 40% Freier Software von Programmierern im Hauptberuf entwickelt.

Nachdem nun die Begrifflichkeiten definiert sind, soll als nächstes auf die Abhängigkeiten der Politik und der Wirtschaft von Software eingegangen werden. Insbesondere soll auf die Gefahren eingegangen werden, die für diese Gesellschaftsbereiche von unfreier Software ausgehen.

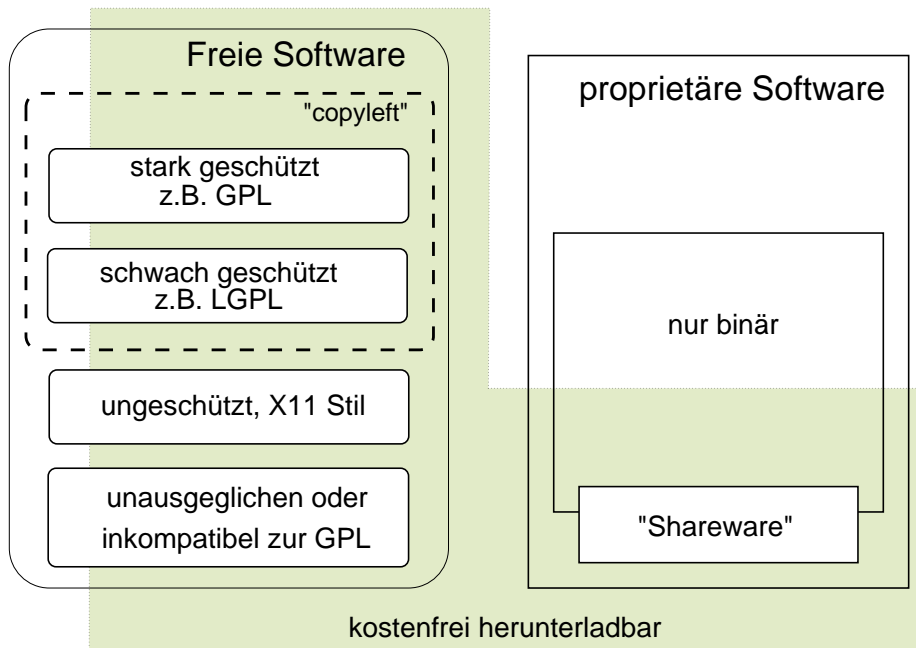


Abbildung 2: Software Kategorien nach (Reiter 2004, S. 86)

4 Bedeutung von Software

Früher waren Computer große Ungetüme und jeder war sich bewußt, wenn er mit ihnen in Berührung kam. Heute begegnen uns Computer nicht immer in der Form, in der wir das erwarten würden. Sie sind so klein, dass sie in nahezu jeden Gegenstand passen. Deshalb ist es schwer, Computer auch immer als solche zu erkennen. Das Mobiltelefon, der Wecker, die Kasse am Supermarkt, das Überweisungsterminal, der Aufzug oder der Fahrkartenautomat; Computer und damit zwangsläufig Software, sind heute allgegenwärtig. Es ist nahezu unmöglich, sich ihnen zu entziehen. Die Auswirkungen dessen auf das persönliche Leben dürften uns bewußt werden, wenn wir uns vor Augen führen was passiert wäre, wenn die Computer nicht funktioniert hätten. Was wäre passiert, wenn die Software die Kreditkarte beim Einkaufen nicht angenommen hätte, das Mobiltelefon sich nicht mit neuem Guthaben hätte aufladen lassen, der Wecker nicht geklingelt hätte oder sich das Überweisungsterminal geweigert hätte, die Überweisung auszuführen?

Nun soll gezeigt werden, dass Software nicht nur unser persönliches Leben beeinflusst, sondern auch die Politik und die Wirtschaft ganzer Länder.

4.1 Politische Auswirkungen

Zuallererst soll die Illusion genommen werden, dass Computer ihrem Anwender gehorchen. Oder haben wir unserem Computer schon einmal den Befehl geben, abzustürzen oder einen Virus zu installieren? Computer gehorchen immer nur dem Computerprogramm, dessen Regeln im Quellcode niedergeschrieben sind. Quellcode und Gesetze sind beides Regulatoren (vgl. Lessig 2004). Code kann jedoch ein noch weitaus effektiverer Regulator sein; er kann nämlich, im Gegensatz zu Gesetzen, nicht ignoriert werden.

Dies soll anhand eines Beispiels verdeutlicht werden. Stellen wir uns vor, wir stehen an einer Verkehrsampel, die Ampel ist rot und wir warten. Auf der gegenüberliegenden Straßenseite stürzt ein Fahrradfahrer und bleibt regungslos auf dem Boden liegen. Weit und breit ist niemand außer uns und dem Verletzten zu sehen. Vermutlich würden wir uns vergewissern, dass die Straße passierbar ist und dem Verletzten Hilfe leisten. Wir hatten uns dazu entschlossen, die rote Ampel als zweitrangig zu betrachten, da wir das Leben eines Mitmenschen in Gefahr sahen. Jeder würde verstehen, warum wir so gehandelt haben. Im schlimmsten Fall würden wir uns vor Gericht rechtfertigen müssen und evtl. eine Strafe für das Vergehen bekommen. Wir konnten uns jedoch dafür entscheiden, diese Regel zu mißachten.

Wäre dies in einem virtuellen Raum geschehen und wäre das Gesetz, dass rote Fußgängerampeln nicht überquert werden dürfen, in Code implementiert gewesen, so hätte es keine Möglichkeit gegeben, dieses Gesetz zu ignorieren. Vielleicht hätten uns die Beine bei dem Versuch, die Straße zu überqueren, einfach nicht gehorcht, vielleicht wären die Füße auch einfach an dem Bürgersteig festgeklebt. In einem virtuellen Raum gelten weder nationale noch physikalische Gesetze; hier gelten die Gesetze, welche der Autor der Software in dieser implementiert hat. Und diese finden ohne Ausnahme Anwendung.

Bei nationalen Gesetzen besteht die Möglichkeit, diese in irgendeiner Form nachzulesen. Da bei unfreier Software der Quellcode, in den meisten

Fällen, nicht zur Verfügung gestellt wird, haben wir keinerlei Möglichkeit herauszufinden, welche Gesetze in der Software implementiert sind. Und selbst, wenn wir die Gesetze kennen, dürfen sie nicht geändert werden. Diese Einschränkung gilt für alle, auch für Regierungen.

So könnte folgender Fall auftreten: in einem Land hatten bisher die zwei größeren Religionsgemeinschaften den Vorteil, dass der Staat für sie, von ihren Mitgliedern, das Geld einbezog. Dazu musste den Einwohnermeldeämtern die Angabe gemacht werden, ob man einer der beiden angehört, oder nicht. Diese Angabe wurde dann in die Datenbank eingetragen. Nach einer sehr kontroversen Diskussion wurde von der Regierung beschlossen, dass das Land in Zukunft auch für die anderen Religionsgemeinschaften die Beiträge der Mitglieder einziehen wird. Dazu muss aber die Datenbank so geändert werden, dass in der Eingabemaske auch die anderen Religionsgemeinschaften zur Auswahl stehen. Die Regierung wendet sich mit diesem Auftrag an den Hersteller der Software. Dieser ist jedoch der Ansicht, dass es sich bei den kleineren Religionsgemeinschaften um „Sekten“ handelt und lehnt es aus Überzeugungsgründen ab, die Änderungen an der Software vorzunehmen. Die Regierung muss nun entweder eine komplett neue Software in Auftrag geben, oder sie kann dieses Gesetz mangels benötigtem Werkzeug nicht durchsetzen. Hierbei ist auch noch die Gefahr der Erpressung denkbar. Ist ein Gesetz für das weitere Fortbestehen einer Regierung von zentraler Bedeutung, könnte der Hersteller auf die Idee kommen, die Anpassung der Software an politische Gegenleistungen zu knüpfen.

Wie wir gesehen haben, sind Regierungen, wie alle anderen Benutzer unfreier Software auch, von Software abhängig, überwacht und kontrolliert. Die Auswirkungen von Software reichen jedoch noch darüber hinaus. Dies werden wir im nächsten Abschnitt sehen, welcher sich mit den wirtschaftlichen Auswirkungen von Software beschäftigt.

4.2 Wirtschaftliche Auswirkung

Unfreie Software führt immer zu einer Monopolbildung. Warum dies so ist, lässt sich wie folgt erklären³:

Geschäfte erfordern Kommunikation mit Kunden/Anbietern.

Um Geschäfte machen zu können, müssen wir kommunizieren. Der Fabrikant möchte gerne wissen, zu welchem Preis er beliefert werden kann und der Verkäufer will einem interessierten Kunden ein Angebot schicken. Ohne mit anderen Menschen zu kommunizieren sind Geschäfte nicht möglich.

Für Kommunikation ist Software erforderlich. Software ist überall, sei es in Mobiltelefonen, Faxgeräten oder natürlich in Personal Computern. Kommunikation ohne Software ist heute undenkbar.

Die menschliche Gesellschaft definiert sich über Kommunikation; die Menschen sind untereinander in einem Maße vernetzt, das über das intuitive Verständnis hinausgeht⁴. Ein Großteil der Kommunikation und nahezu die gesamte Wirtschaft ist heute vollständig von Software abhängig. Alleine betrachtet, wäre dies für die Wirtschaft nicht besonders bedenklich. Jedoch besitzt unfreie Software die folgende Eigenschaft:

Unfreie Software funktioniert nur mit sich selbst gut. Viele Leute wissen es aus Erfahrung; wenn sie ein Textdokument erhalten haben, dass mit einem bestimmten Textverarbeitungsprogramm geschrieben wurde, benötigen sie meist dasselbe Programm, um das Dokument fehlerfrei zu betrachten. Oft ist sogar die gleiche Version der Textverarbeitung nötig.

Dieses Verhalten ist bei unfreier Software auch nicht weiter verwunderlich. Der Hersteller von unfreier Software hat gar kein Interesse daran, dass dem anders wäre. Er will, dass möglichst viele Leute seine Software einsetzen müssen und sie gezwungen sind, verfügbare Updates des

³Dieser Abschnitt basiert auf der Argumentation von Georg Greve.

⁴vgl. hierzug Stanley Milgrams „Six degrees of separation“-Experiment in (Barabasi 2003, S. 27-30).

Programmes auch zu kaufen. Unfreie Software erlaubt es zwar standardisierte Dateiformate zu öffnen, speichert man diese jedoch wieder ab, kann die Datei nicht mehr von anderen Programmen gelesen werden, die den Standard befolgen. Dies wird dadurch erreicht, dass zusätzlich zu dem Standard noch Erweiterungen hinzugefügt werden. Somit ist der Anwender gezwungen weiterhin bei der Software des Herstellers zu bleiben⁵.

Freie Software dagegen begünstigt das problemlose Austauschen von Daten, sowie Interaktion zwischen Programmen. Jeder Autor eines neuen Programmes, kann auf Quelltextteile von bereits bestehenden zugreifen und diese für sein eigenes Programm verwenden. Daher ist der Import und Export von Daten in andere freie Formate meist problemlos möglich. So können wir z.B. mit den beiden freien Textverarbeitungsprogrammen Kword und Abiword problemlos OpenOffice.org Textdateien öffnen, bearbeiten und abspeichern.

Betrachten wir nun alle Punkte zusammen, so werden wir feststellen, dass unfreie Software zwangsläufig zu einem Monopol führen wird. Es ist nicht die Strategie einzelner Firmen, die dies bedingt, sondern die Tatsache, dass diese unfreie Software verwenden. Das Fatale daran ist, dass dieses Monopol nicht auf den Softwarebereich begrenzt bleibt. Es breitet sich auch in den Hardwaresektor aus. Wenn wir in ein Computergeschäft gehen, werden wir feststellen, dass Intel Computer fast ausschließlich mit Microsoft Windows Betriebssystem verkauft werden. Umgekehrt läuft Microsoft Windows nur auf Intel-kompatibler Hardware⁶.

Noch tiefgreifendere Auswirkungen auf die Wirtschaft werden deutlich, wenn wir die Studie von (Miller 2004) zur Hand nehmen. Danach sind 50% der deutschen Industrie und 80% der Exporte von der Informations- und Kommunikationstechnologie abhängig. Das bedeutet, dass 50% der deutschen Industrie und 80% der deutschen Exporte den Preis für dieses Monopol be-

⁵Diese Praktik ist weit verbreitet und wird als „Vendor Lock-In“ bezeichnet.

⁶Dieses Monopol wird daher oft auch als „Wintel“ Monopol bezeichnet.

zahlen⁷. Deutschland und andere Industrieländer können sich diesen Preis noch leisten, Entwicklungsländer nicht (vgl. Heinz 2001).

Bei Freier Software kann kein schädliches „Kontrollmonopol“ entstehen. Wenn ein bestimmtes freies Softwareprogramm nahezu 100% Marktanteil hätte, so ist dies lediglich ein „Funktionsmonopol“. Dieses birgt aber keine künstlichen Einstiegsbarrieren für Konkurrenten. Jeder, der genügend Zeit investiert, könnte das gleiche Wissen wie der Originalautor der Software erreichen und Dienstleistungen für diese Software anbieten. Bei unfreier Software kann nur der Hersteller Dienstleistungen wie Anpassungen, Fehlerbehebungen und Sicherheitsupdates durchführen. Außerdem hat er die Kontrolle darüber, welche anderen Programme mit seinem funktionieren können und welche nicht.

Beispielsweise wurde Microsoft von der Europäischen Kommission für schuldig befunden, wettbewerbswidrige Methoden einzusetzen⁸. Microsoft versuchte, seine Monopolstellung im Bereich der Software für Arbeitsplatzrechner zu nutzen, um auch die Kontrolle über den Servermarkt⁹ zu erlangen. Dies hätte dem Softwaremarkt und dadurch wiederum der gesamten europäischen Wirtschaft starken Schaden zugefügt.

5 Rolle der Organisationen

Organisationen müssen für ihre Arbeit mit Menschen, aus Entwicklungsländern, kommunizieren. Und wie bereits oben erwähnt, findet Kommunikation heute meist über Software statt: es müssen Dokumente miteinander ausgetauscht, von beiden Seiten geöffnet, bearbeitet und wieder abgespeichert werden können. Benutzt eine der beiden Seiten unfreie Software, so muss die andere zwangsläufig nachziehen, ansonsten können sie nicht mehr ohne

⁷Des weiteren kommt die Studie zu dem Ergebnis, dass „viele Bereiche der öffentlichen Verwaltung, des Bildungs- und Gesundheitswesens“ sich durch Software effizienter gestalten lassen und Kosten in Milliardenhöhe eingespart werden können (Miller 2004, S. 10).

⁸Microsoft hat dagegen vor dem Europäischen Gerichtshof Klage eingereicht. Das Verfahren dazu läuft derzeit noch.

⁹Ein Server ist ein Programm, welches Dienste für andere Programme (Client) anbietet. Client und Server kommunizieren meist, mit Hilfe eines bestimmten Protokolls, verschlüsselt miteinander.

Probleme miteinander arbeiten.

Mehrfach diskutierte die Zivilgesellschaft beim WSIS über die mögliche Einführung eines Videokonferenzprogrammes¹⁰. Dafür wurde, aus technischen Gründen, auch immer wieder unfreie Software in Betracht gezogen¹¹. Wäre dies beschlossen worden, hätte es fatale Auswirkungen gehabt. Will man an der Diskussion teilnehmen, muss man, mit allen Konsequenzen die daraus folgen, unfreie Software einsetzen. Entscheidet man sich dagegen, wird man von der Diskussion, die über die eigene Zukunft entscheidet, ausgeschlossen.

Diese Ausgrenzung ist bei weitem kein Einzelfall. Dies erkennen wir jedoch erst, wenn wir selbst konsequent Freie Software einsetzen. Nur dann bemerken wir, dass beispielsweise UNO Dokumente oft nicht mit Freier Software zugänglich sind.

6 Fazit

Wie wir gesehen haben, ist Software heute allgegenwärtig. Egal was wir auch tun, wir kommen fast immer mit Computern in Berührung und wir sind auf diese angewiesen. Da Computer nicht dem Anwender, sondern Computerprogrammen gehorchen, werden wir von diesen eingeschränkt. Haben wir nicht die Freiheit, die Software an unsere eigenen Bedürfnisse anzupassen, oder von anderen anpassen zu lassen, sind wir hilflos.

Die Regierung hat, wenn sie unfreie Software einsetzt, nicht mehr die Kontrolle über ihre eigenen Werkzeuge. Sie ist dadurch abhängig und erpressbar, was demokratisch sehr bedenklich ist.

Benutzt ein Entwicklungsland unfreie Software, erschwert es sich dadurch die Möglichkeit, eigene Kompetenzen in der Softwareentwicklung aufzubauen. Nahezu die gesamte Wirtschaft eines Landes ist jedoch heutzutage auf die Informations- und Kommunikationsindustrie angewiesen. Das bedeutet,

¹⁰Die meisten dieser Diskussionen fanden auf der WSIS CS Plenary Mailingliste, archiviert auf <http://mailman.greenet.org.uk/public/plenary/>, unter den Betreffs „Collaboration software debate“ und „Virtual Participation“ statt.

¹¹Es ist alleine schon fraglich, ob Videokonferenzen ein geeignetes Werkzeug für diese Art von Diskussion sind. Darauf soll hier jedoch nicht näher eingegangen werden.

dass ein Land ohne eigene Softwareindustrie wirtschaftlich vollständig vom Ausland abhängig ist.

Entwicklungsländern haben durch Freie Software die Möglichkeit in diesem Bereich, wirtschaftlich und politisch, eigenverantwortlich zu handeln. Bereits bestehende Komponenten können, von einheimischen Firmen, an lokale Bedürfnisse angepasst werden. So könnte die Regierung es z.B. für wichtig erachten, Minderheiten den Zugang zum öffentlichen Dienst zu erleichtern. Dazu könnte sie ihre Computerprogramme, entweder selbst oder von einem Unternehmen, in deren Sprachen übersetzen lassen. Bei unfreier Software sind wir auf den Hersteller angewiesen. Ist es für diesen nicht gewinnbringend, was oft der Fall ist, wird keine Anpassungen gemacht.

Freie Software ermöglicht es der Regierung¹², sowie der Wirtschaft, ihre Aufgaben zu erledigen, ohne vom Ausland abhängig zu sein. Ein weiterer positiver Effekt: investiert die Regierung z.B. in eine neue Verwaltungssoftware, so zirkuliert dieses Geld im eigenen Land und fördert dadurch gleichzeitig die eigene Wirtschaft.

Wenn wir selbst konsequent Freie Software einsetzen, merken wir es, wenn anderen der Zugang zu Informationen verwehrt wird, die für deren weitere Entwicklung essenziell sind. Des weiteren können wir mit anderen kommunizieren, ohne dass wir diese in ihrer eigenen Softwarewahl beeinflussen. Dadurch ermöglichen wir es Entwicklungsländern, ohne große Probleme, selbst Freie Software einzusetzen. Nur so können sie die Kulturtechnik Software, in ihrem ganzen Umfang nutzen. Dies wiederum schafft die Basis für eine nachhaltige Entwicklung.

¹²Regierungen sind im Regelfall der größte Softwarekunde in einem Land.

Quellenangaben

- [Barabasi 2003] Barabasi, Albert-Laszlo (2003): *Linked. How Everything Is Connected To Everything Else and What It Means for Business, Science, and Everyday Life*. Plume (Penguin Group), New York / London.
- [Gay 2002] Gay, J. (Hrsg.) (2003): *Free Software Free Society: selected Essays of Richard M. Stallman*. Erste Ausgabe, GNU Press 2002, Boston; <http://www.gnupress.de>
- [GNU's Bulletin 1987] Free Software Foundation: *GNU's Bulletin*, Vol. 1 No. 6, June 1987, Boston; <http://www.gnu.org/bulletins/bull6.html> vom 30.06.2005.
- [Grassmuck 2002] Grassmuck, Volker (2002): *Freie Software zwischen Privat- und Gemeineigentum*, Bundeszentrale für politische Bildung, Bonn.
- [Greve 2003] Greve, Georg C. F. (2003): *Fighting Intellectual Property: Who Owns and Controls the Information Societies?*, in: Heinrich Böll Foundation (ED.): *Visions in Process, World Summit on the Information Society - Geneva 2003, Tunis 2005*.
- [Heinz 2001] Heinz, Federico und Oscar E. Heinz (2001): *Proprietary Software and Less-Developed Countries - The Argentine Case*; <http://www.vialibre.org.ar/index.php/article/articleprint/15/-1/12/> vom 30.06.2005.
- [Lakhani et al. 2002] Lakhani, Karim R. und Bob Wolf und Jeff Bates (2002): *Boston Consulting Group, Hacker Survey, Release 0.3*; <http://www.bcg.com/opensource/BCGHACKERSURVEY.pdf> vom 30.06.2005.
- [Lessig 2004] Lessig, Lawrence (1999): *Code And Other Laws Of Cyberspace*, New York, NY.

- [Miller 2004] Miller, Franz (2004): Innovationstreiber Informations- und Kommunikationstechnik, Fraunhofer Magazin 2.2004; http://www.fraunhofer.de/fhg/Images/mag2-2004-08_tcm5-9201.pdf vom 30.06.2005.
- [Reiter 2004] Reiter, Bernhard E. (2004): Wandel der IT: Mehr als 20 Jahre Freie Software, in: H. Sauerburger (Hrsg.): Praxis der Wirtschaftsinformatik HMD 238, S. 83-91.