UNCTAD-ICTSD Project on IPRs and Sustainable Development

Intellectual Property and Computer Software

A Battle of Competing Use and Access Visions for Countries of the South

By Alan Story
Lecturer in Intellectual Property Law, University of Kent, United Kingdom

ICTSD
International Centre for Trade and Sustainable Development

UNCTAD

Issue Paper No. 10

May 2004 | Intellectual Property Rights and Sustainable Development

UNCTAD–ICTSD Project on IPRs and Sustainable Development

Intellectual Property and Computer Software

A Battle of Competing Use and Access Visions for Countries of the South

By Alan Story

Lecturer in Intellectual Property Law, University of Kent, United Kingdom

ICTSD

International Centre for Trade and Sustainable Development

UNCTAD

Issue Paper No. 10

Published by

International Centre for Trade and Sustainable Development (ICTSD)
International Environment House
13 chemin des Anémones, 1219 Geneva, Switzerland
Tel: +41 22 917 8492 Fax: +41 22 917 8093
E-mail: ictsd@ictsd.ch Internet: www.ictsd.org

United Nations Conference on Trade and Development (UNCTAD)
Palais des Nations
8-14 avenue de la Paix, 1211 Geneva 10, Switzerland
Tel: +41 22 907 1234 Fax: +41 22 907 0043
E-mail: info@unctad.org Internet: www.unctad.org

Funding for the UNCTAD-ICTSD Project on Intellectual Property Rights and Sustainable Development has been received from the Department of International Development (DFID, UK), the Swedish International Development Agency (SIDA, Sweden) and the Rockefeller Foundation.

The Project is being implemented by the International Centre for Trade and Sustainable Development (ICTSD) and the secretariat of the United Nations Conference on Trade and Development (UNCTAD) (Project Number INT/OT/1BH). The broad aim is to improve the understanding of intellectual property rights-related issues among developing countries and to assist them in building their capacity for ongoing as well as future negotiations on intellectual property rights (IPRs).

For details on the activities of the Project and all available material, see
http://www.iprsonline.org/unctadictsd/description.htm

Printed on Cyclus Print 100% recycled paper by Imprimerie Typhon,
41rte de la Fruitière, 74650 Chavanod, France. April 2004

ISSN 1681-8954

CONTENTS

The author would be glad to hear your comments or criticism on this report.
Please contact: Alan Story, Kent Law School, University of Kent, Canterbury,
Kent CT2 7NS, United Kingdom, (acs3@kent.ac.uk).

FOREWORD

This paper is a contribution of the joint UNCTAD-ICTSD Project on Intellectual Property Rights and Sustainable Development to the ongoing debate on the impact and relevance of intellectual property rights on new software technologies and development. In particular, it examines the conceptual, economic and intellectual property law implications of 'proprietary' and 'free' software formats.

The paper puts forward five main findings. First, that copyrights and other forms of intellectual property protection have erected a clear barrier to the spread of software across the South. Second, free software formats are moving fast in most developing countries where users are attempting to develop new products, innovations and adaptations in a effort to reduce the digital divide. Third, after comparing the license costs of proprietary and free software formats, the paper suggests that costs associated with free software are significantly lower although some learning, maintenance and services costs need to be taken into account when adopting these technologies. Fourth, free software formats might offer different advantages for technology transfer and follow-up applications depending on the model used (i.e. open source or general public licences). Finally, in the current international context free software systems are not a mere policy choice for developing countries, they are an important alternative for building, maintaining and changing rules that govern information flows.

Intellectual property rights have (IPRs) never been more economically and politically important or controversial than they are today. Patents, copyrights, trademarks, industrial designs, integrated circuits and geographical indications are frequently mentioned in discussions and debates on such diverse topics as public health, food security, education, trade, industrial policy, traditional knowledge, biodiversity, biotechnology, the Internet, the entertainment and media industries. In a knowledge-based economy, there is no doubt that an understanding of IPRs is indispensable to informed policy making in all areas of human development.

Intellectual property was until recently the domain of specialists and producers of intellectual property rights. The TRIPS Agreement concluded during the Uruguay Round negotiations has signalled a major shift in this regard. The incorporation of intellectual property rights into the multilateral trading system and its relationship with a wide area of key public policy issues has elicited great concern over its pervasive role in people's lives and in society in general. Developing country members of the World Trade Organization (WTO) no longer have the policy options and flexibilities developed countries had in using IPRs to support their national development. But, TRIPS is not the end of the story. Significant new developments are taking place at the international, regional and bilateral level that build on and strengthen the minimum TRIPS standards through the progressive harmonisation of policies along standards of technologically advanced countries. The challenges ahead in designing and implementing IP-policy at the national and international levels are considerable.

Empirical evidence on the role of IP protection in promoting innovation and growth in general remains limited and inconclusive. Conflicting views also persist on the impacts of IPRs in the development prospects. Some point out that, in a modern economy, the minimum standards laid down in TRIPS, will bring benefits to developing countries by creating the incentive structure necessary for knowledge generation and diffusion, technology transfer and private investment flows. Others stress that intellectual property, especially some of its elements, such as the patenting regime, will adversely affect the pursuit of sustainable development strategies by raising the prices of essential drugs to levels that are too high for the poor to afford; limiting the availability of educational materials for developing country school and university students; legitimising the piracy of traditional knowledge; and undermining the self-reliance of resource-poor farmers.

It is urgent, therefore, to ask the question: How can developing countries use IP tools to advance their development strategy? What are the key concerns surrounding the issues of IPRs for developing countries? What are the specific difficulties they face in intellectual property negotiations? Is intellectual property directly relevant to sustainable development and to the achievement of agreed international development goals? Do they have the capacity, especially the least developed among them, to formulate their negotiating positions and become well-informed negotiating partners? These are essential questions that policy makers need to address in order to design IPR laws and policies that best meet the needs of their people and negotiate effectively in future agreements.

It is to address some of these questions that the joint UNCTAD-ICTSD Project on Intellectual Property and Sustainable Development was launched in July 2001. One central objective has been to facilitate the emergence of a critical mass of well-informed stakeholders in developing countries - including decision makers, negotiators but also the private sector and civil society - who will be able to define their own sustainable human development objectives in the field of IPRs and effectively advance them at the national and international levels.

[illegible]

Ricardo Meléndez-Ortiz
ICTSD Executive Director

R Ricupero

Rubens Ricupero
UNCTAD Secretary General

EXECUTIVE SUMMARY

This report was commissioned to examine a range of economic, political and developmental issues connected with the use and expansion of computer software in countries of the South. In particular, it examines the wider conceptual, economic and intellectual property law implications and practical consequences of the two main software formats: the still predominant proprietary format, best exemplified by the operating and application programs of Microsoft Corp. and the increasingly important open source and free software formats, grouped under the acronym, FLOSS (Free/Libre/Open Source Software) The main question it attempts to analyse and answer is this: both in the short- and long-term, which format best meets the need of the less industrialised countries of the South? The principal conclusion drawn is that, for a wide range of reasons, free and open source software is the preferred alterative for countries of the South.

This report is aimed at the layperson who does not have a sophisticated background in computing and avoids the use of technical computer language whenever possible; indeed, the report was written with a non-technical (and non legally-trained) audience in mind. An Appendix provides definitions and simple explanations of seven key computing terms.

Section 1 provides a brief introduction to the entire report and the issues raised in its six sections. It gives an overview of the growing 'software war' between the two formats and their proponents, provides some financial statistics on the global sales (and licensing) of computer software, and argues that software issues are too important to be left exclusively to software companies or software programmers.

The next three sections give an explanatory orientation to the heart of this report, which is found in Section 5. Section 2 provides a brief introduction to the two main types of software and their characteristics; it focuses on their contrasting approaches to the protection and use of the source code, the internal programming language of software. Section 3 examines the three main branches of intellectual property law — copyright, patents, and trade secrets — that are implicated in the legal protection (and restriction) of software. The legal history of these regimes vis à vis software is briefly outlined, as are the legal requirements for protection mandated by the Agreement on Trade-Related Aspects of Intellectual Property Rights 1997 (TRIPS Agreement). This section commences with a short contextualisation of how legal rights, such as intellectual property rights, should be understood, particularly their contingency. Section 4 attempts to give a contemporary 'snapshot' of some of the leading — as well as more typical — FLOSS projects and developments across ten countries of the South, including China, India, Thailand, South Africa, Uganda, Namibia, Lebanon, Mexico, Brazil and Peru; although the future of FLOSS developments in these and similar countries is hardly assured, there have been a number of path-breaking advances.

The heart of this report is found in Section 5. Six inter-linked issues are analysed; they include: software costs and licensing, computer hardware requirements for the two competing software types, technology transfer and software (a key question for the future economic growth of the South), whether the South should mimic legal developments, particularly in the United States, that permit the patenting of software and whether software patenting assists or hinders software innovation, which form of software format creates the best employment opportunities in the South, and a long section on unauthorised software copying (often misleadingly called 'software piracy'). Again, the analytical framework used is a comparison of the two formats; the conclusions from a number of leading studies and anecdotal evidence are presented and assessed.

The final part, Section 6, highlights a few of the critical issues that will impact on the use and further spread of software across the South; in particular, it emphasises, to quote from one recent study, that which software is employed is not merely a matter of mere 'product choice' and that free and open source software "reflects more fundamentally an alternative strategy for the building, maintaining and changing the rules that govern information flows in the economy."

1. INTRODUCTION

The information age; The computer era; The digital revolution; Although these are much hyped and sometimes exaggerated phrases, especially given the global disparity in information technology that exists between industrialised Northern countries and countries of the South[1], it is no exaggeration to say that truly wonderful communication possibilities have opened up in the past two decades.(Okediji) Almost on a weekly basis, new technologies are being created that hold out potentially transformative and more inclusive ways to communicate, to teach and to learn, to compute and organise data, to conduct business, to promote democratic dialogue and improved governance — and to organise resistance to injustice and oppressive governance. All who wish to participate in this global information age, however, need to have access to a computer. And such computers, the hardware component, require software for their operation. This software component exists in two basic formats or forms: the proprietary form, of which the best known example is the Windows computer operating systems of the US Microsoft Corporation, and FLOSS, Free/ Libre / Open Source Software.[2] This report focuses primarily on which type of software is the best form, now and in he foreseeable future, for the needs of countries of the South, examines the legal regimes protecting (and restricting) software, and tries to place software within the wider developmental agenda of such countries.

While certainly trying to avoid technological determinism, this report takes the view, as a starting point, that the question of what is the best form of computing software is far too important a matter to be left to Bill Gates, chairman of Microsoft, or, for that matter, to the programmers, 'geeks' and 'hackers' — the terms are used here affectionately — from the free and open source software movements. Not only are a number of important economic issues at stake — for individual computer users, for software writers and programmers, for institutions and organisations providing computers, and for whole countries, especially those in the South — but so, too, is a range of complex political, social, and philosophical issues. As one organisation of computer professionals commented:

"Tomorrow's information and communication infrastructure is being shaped today. But by whom and to what ends? Will it meet the needs of all people? Will it help the citizenry address current and future issues? Will it promote democracy, social justice, and sustainability? Will the appropriate research be conducted? Will equitable policies be enacted?" (CPSR 2002)

As is examined in more detail in Sections 5 and 6, it is argued here that governments in the South and those inside and outside governments who advise them (including commentators in the international development community, and those who lobby governments, such as NGOs) need to join into this debate as well. Already at least three countries, China, South Africa and, to a lesser extent India, have made important decisions in this debate (Section 4). The *Idlelo: First African Conference on the Digital Commons*, held in Cape Town, South Africa in January 2004 and attended by this author, testifies to the energy and dynamism of the free and open source movement on that conference; two and a half months later, the conference follow-up e-mail discussion list remains very active and informative.

The international software debate, which is growing increasingly heated[3] and labelled the 'software wars' by some commentators, is occurring on several planes. One dimension is ideological and embraces radically different views of the role and purpose of software. Is it merely another commodity that should be restrictive, closed to user modification, and licensed from behind the walls of the triple fortresses called copyright, patent and trade secret law? Alternatively, commentator Tony Stanco has called the drive for free software "an international social movement that touches on the fundamental human rights of freedom and democracy."(Scheeres) A second dimension is the legal one. It is precisely the legal barriers erected by trade secret, copyright, and patent law, first domestically in rich industrialised countries and, more recently, globally by the actions of their governments (e.g. in the provisions of the 1994 Agreement on Trade-Related Intellectual Property Rights (TRIPS)) that have been one of the key sparks behind the growth and acceptance of FLOSS, including in countries of the South. The highlights and wider implications of these legal regimes are included here.

Due to time and financial constraints, it has not been possible in this study to collect and analyse detailed statistics on the economic impact of software on countries of the South. On several occasions, such as in Section 5.1, this report does refer to comparisons, including some calculated during the course of this study's research, between proprietary software and FLOSS for particular computing projects. But if the macro financial impact and consequences of current software costs (either direct or indirect) for countries of the South are not fully calculable for the moment, some sense of the wider global financial stakes can be gleaned from the recent reports on what are labelled the 'copyright industries' in the United States, the world's largest producer and exporter of copyright-protected items …or intangible 'expressions', to use the copyright law term of art. Taking into account all of the leading US copyright industries, which includes those producing films, television programs, audiovisuals works, music (CDs, records and cassettes), and books, periodicals, newspapers and other publications in print and digital format, "foreign sales and exports of computer software have consistently generated *the highest dollar value and fastest growth,* rising from $[US] 19.65 billion in 1991 to $[US] 60.74 billion in 2001."(Siwek, italics added) By comparison, foreign revenues in 2001 from the sales of US films were $US 14.69 billion, from the pre-recorded record and tape industry, they were $US 9.51 billion, and from the combined publishing industries, the total was $US 4.03 billion. In other words, the value of foreign sales of software was more than twice the *combined* value of the entire US film, music and publishing industries. Clearly most of these software 'sales' — actually licensing revenues in many cases — were made to other rich countries, particularly in Europe; such countries have much higher per capita income levels and far greater levels of per capita computer ownership and usage and Internet access (ITU 2002). Yet, countries of the South generally and especially countries with large populations, such as China, India, Indonesia, Brazil, and Nigeria, offer huge potential markets for the intellectual property-protected software produced in rich northern countries. Indeed, countries of the South generally hold out the promise of becoming key consumers of proprietary software.[4] This means that there is a very large financial inventive for corporate interests in rich countries, especially the United States, to require countries of the South to provide (as is documented in Section 3) the strongest possible protection to intellectual property-protected products, including software, within their own borders: the products protected will primarily be of US origin and/or ownership. This conclusion raises the question: is the use of FLOSS one avenue for countries of the South to escape from this expensive proprietary protectionism?

The software world is a rapidly changing world and this report attempts, in Section 4, to give a snapshot view of the contemporary software world, especially of FLOSS, across countries of the South in 2003 and early 2004. Although proprietary software still holds the predominant position, numerous reports from large commercial intelligence reporting agencies mirror the conclusions of a recent Deutsche Bank Research publication. It noted:

"The freely available operating system Linux — a so-called open-source program — has reached a level of sophistication putting it at least on a par with the quality of its proprietary rivals by Microsoft, Sun and other providers…Interest in open source is growing rapidly in industry and government agencies in response to the demonstrable cost benefits and presumed advantages in terms of stability and security. Although Linux's market share is small, it is advancing fast." (Deutsche Bank)

Meanwhile, planning documents prepared for the recently-held World Summit on the Information Society in Geneva concluded that to promote improved "access to information and knowledge", the "development and deployment of open-source software and standards for ICT [information and computing technology] networking should be encouraged."(World Summit on the Information Society) Several years ago, a prominent business magazine noted that "[a]head of the crowd, Bill Gates located the sweet spot in the business of bits and bytes; as a provider of a 'platform', Windows is essentially a collection of building blocks that developers need to create applications."(Economist, 18 October 2001). In the all-important operating system market, that is, where Windows and Linux face each other as head-to-head rivals (as in shown in Section 4), that 'sweet spot' has, of late, become a bitter pill for many and been subject to increasing scepticism and challenge across the globe. China's recent enthusiasm for open source will provide Microsoft with a formidable challenge and the overall future of software — that is, which type, if any, will become the global standard — is far from pre-determined on any continent, even including the United States itself.

To date, the overwhelming focus of research on FLOSS has been on 'typical projects', break-through developments, and debates in the industrialised world; this is where, at least until quite recently, most higher profile FLOSS projects have been located. Yet, this pattern is starting change as more and more commentators, including in the mainstream commercial media, have begun to appreciate the possibilities of FLOSS in the South. As one publication recently put it, it's "…even more electrifying effect in the developing world … (where such software) holds out the promise of high-tech independence."(Newsweek, June 2003) Still it does need to be appreciated that few reliable conclusions about software development in the South can be drawn by trying to directly extrapolate the lessons from a software development project established in a country such the United States or Germany. Without even looking at the widely different cultural contexts, the technical conjuncture is obviously also very different. In the United States, for example, recent survey data show that 5,375 persons out of 10,000 persons are Internet users and there are an estimated 62.50 personal computers per 100 inhabitants.(ITU Report, 2000) In Brazil by comparison, 822 persons out of 10,000 are Internet users and there are 7.48 per computers per 100 inhabitants. In a much poorer country of the South such as Zambia, there are a mere 49 persons per 10,000 who are Internet users and a very small number of computers, only 0.75 per 100 Zambians.[5] And because FLOSS is still such recent phenomenon in the South, key distinctions between proprietary software and FLOSS are often not drawn. For example, a very detailed research study in 2001 on the costs of computers across a range of various school types in South Africa and Zimbabwe assumed that all such school computers would be operated by proprietary software operating system, that is, Microsoft Windows. Still, there are some encouraging signs that recently announced research projects may start to provide at least a more reliable empirical basis for the software debates. In January 2003, Bridges.org., an international NGO based in Cape Town, South Africa announced that it was launching a two-year study which will examine the implications of choosing either of the two software types for South Africa and Namibia; an initial report is expected shortly. "The debate over which kind of software is better for the developing countries is heating up and many argue that the choice will have long-term implications as African countries take steps to join the information society," Bridges explained in making this announcement.(Otter, 2003) The International Centre for Trade and Sustainable Development in Geneva has also shown its interest in this issue by commissioning the report that you are now reading. The types of software that are used to operate computers certainly raise a number of sustainable development issues across the globe.

Here is how this report is organised: Sections 2 and 3 provide background to the analysis of, respectively, the two main types of software and the various legal regimes, domestically and internationally, that provide intellectual property protection to computer software. Section 4 give a flavour or snapshot of the contemporary software situation across countries of the South and the increasingly hard-fought conflict that is emerging between traditional closed source proprietary software, particularly Microsoft products, and a range of FLOSS programs. Section 5, the main part of this report, compares and contrasts proprietary and FLOSS software and looks a spectrum of issues, such as costs, technology transfer, employment and training, patents and innovation, and the unauthorised use of software (or software 'piracy'). Given the critical importance of educational concerns for countries of the South, software issues in schools and universities are a particular, though not exclusive, focus. Section 6 includes some brief commentary on the future and the policy issues and implications that lie ahead. Appendix 1 provides definitions of seven key technical terms related to computer software that the average reader needs to appreciate in reading this report.

This is very much an introductory study and attempts to put an issue on the global development agenda which, to date, has generally received limited attention. Technical computer language has been reduced to a minimum; indeed, the report was written with a non-technical audience in mind. Some of the material in this report appeared earlier in a different form in 2002 in a report this author prepared for the UK Commission on Intellectual Property Rights, though, given the fast-moving nature of the software world, certain matters have moved on, sometimes quite dramatically, in the past 18 months.(Story, 2002)

2. THE TWO MAIN TYPES OF SOFTWARE[6]

For the purposes of this report, the most helpful way of categorising computer software (vis à vis ownership, use patterns and possibilities, and their intellectual property implications) is to divide it into two categories: (a) Proprietary software; (b) FLOSS (Free / Libre / Open Source Software)

However, certain distinctions can also be drawn between somewhat different types of FLOSS: Free Software and Open Source Software. (OSS)

2.1 Proprietary Software

As its name implies, proprietary software is software that is owned as private property by a company (or occasionally by an individual software developer). Its 'private propertiness' is protected by various intellectual property laws and regimes, as is explained in more detail in Section 3, and by the licences required for its use.

How to protect and or not protect and use the source code (the internal programming language) of software is at the heart of most legal, policy, and practical debates about software. The issue encompasses both operating system programs (e.g.Windows), which manage the internal function of the computer, and application programs (e.g. Microsoft Word, games, spread sheets and other word processing programs), which perform specific data processing tasks for users. The code of a program is what makes it particularly valuable and transforms it — potentially at least — into a creative tool that can be used to solve a range of problems and acts as a catalyst or building block for further developments and new applications. In other words, the source code is what makes software a 'living', adaptable technology capable of improvement and modification and not simply a fixed and a pre-packaged, non-adaptable computing solution. Of course, the 'average' computer user is not interested in either gaining access to the source code of a program or learning how to edit that code. But many others are interested, and often must have access to the source code. Those actually operating computer networks and systems, whether in business, government, educational institutions or community organisations, and those with the requisite skills in programming and writing software require source code access and the unrestricted ability to modify it so that purchased or bespoke software can work for their specific needs. With proprietary software, this is not possible — unless special licences or exemptions are granted by the owner — and domestic and international intellectual property laws establish the legal framework that restrict such access.

In the case of proprietary software, these key restrictions are made manifest by the particular licensing terms that those actually licensing or operating the software are required to abide by, again as a matter of law. (Software is seldom sold.) Under the terms of what is known as the 'End User License Agreement', this source code cannot be copied, shared, modified, redistributed, or reverse engineered by other software developers or users. In the same vein and as a key part of the business model of proprietary software, the license will usually permit use of the software on only a single computer and/or require extra licensing fess to be paid for each additional computer or work station using the software (e.g. in schools and colleges where there may be hundreds or even thousands of such computers available for student and staff use). The code used for application programmes, which make up the bulk of computer programmes today, must be compatible with the code found in the operating system. The problem (and the resulting control) is further exacerbated because most hardware systems come 'pre-loaded' with various types of proprietary software; the cost of such software is built into the system's purchase price.

Given the dramatic increase in computer usage over the past decade, at least in the richer industrialised countries, and given the ancillaries that spring inherently from code 'ownership', especially for an operating system such as Windows that has become the world standard, it starts to become clear how closed source and intellectual property-protected computer software can come to represent a significant source of wealth and power. For example, the contract to provide software for the National Health Service in the United Kingdom is worth an estimated £5.0 billion and Microsoft took most of this business; for local governments in the United Kingdom, the software bill is £2.4 billon for this year (2004) alone. (Cross)

2.2 FLOSS

Both types of Free/Libre/Open Source Software share one key characteristic: all users must have open access to the source code, which is considered a shareable and non-'propertised' resource. At the same time, there are certain differences in approach.

Free Software

A report entitled "Free Software / Open Source: Information Society Opportunities for Europe?" gives a useful summation of the main features that characterise free software and it is worthwhile to quote from this report in detail. This alternative approach means that all users have

"the freedom to:

- *Use the software as they wish, for whatever they wish, on as many computers as they wish, in any technically appropriate situation.*
- *Have the software at their disposal to fit it to their needs. Of course, this includes improving it, fixing its bugs, augmenting its functionality, and studying its operation.*
- *Redistribute the software to other users, who could themselves use it according to their own needs. This redistribution can be done for free, or at a charge, not fixed beforehand.*

It is important now to make clear that we are talking about freedom, and not obligation. That is, users of [such a software program]… can modify it, if they feel it is appropriate. But in any case, they are not forced to do so. In the same way, they can redistribute it, but in general, they are not forced to do so.

To satisfy those previous conditions, there is a fourth one, which is basic, and is necessarily derived from them:

- *Users of a piece of software must have access to its source code."*

(Working group on Libre Software).

To facilitate these various freedoms and to make sure that the source code does not become the private or exclusive property of any one particular software developer or a group of developers, the pioneers of the free software movement, and in particular the US computer programmer Richard Stallman, developed in the 1980's what is called the General Public Licence (GPL). Its main purpose is to ensure and reinforce a sharing ethos with the source code of programs such as Linux, the basic free/open source operating system. The Free Software approach to licensing is based on what has become known as the 'Copyleft' principle. Flipping, so to speak, the usual purpose of copyright — exclusivity — on its head, a 'Copyleft' license such as the GPL means that code must be shared with others and does not allow any user to distribute the code and its modifications, enhancements, and additions as part of a proprietary scheme. In addition, the GPL requires that the enhancements be licensed on the same terms as the code which the developer initially received and used.

Open Source Software

But certain other software developers, while believing that computer source code should remain open and accessible, considered that the 'free software' approach did not provide the basis for a commercial business model and established in the 1990s another organisation, the Open Source Initiative (the 'OSI') that operates on some similar and some different principles. OSI defines Open Source as software providing the following rights and obligations:

1. *"No royalty or other fee imposed upon redistribution.*
2. *Availability of the source code.*
3. *Right to create modifications and derivative works.*
4. *May require modified versions to be distributed as the original version plus patches.*
5. *No discrimination against persons or groups.*
6. *No discrimination against fields of endeavour.*
7. *All rights granted must flow through to/with redistributed versions.*
8. *The license applies to the program as a whole and each of its components.*

9. *The license must not restrict other software, thus permitting the distribution of open source and closed source software together.*"
(Open Source Initiative)

In particular, open source software licenses allow software developers exclusive protection to additions that they make to a program; these enhancements provide one of the their key revenue streams. For example, an alternative licensing scheme to the GPL, known as the Debian Social Contract, gives licensees greater flexibility by allowing them to bundle software code that was developed co-operatively with proprietary code. In other words, no restrictions can be placed on other software that is distributed along with the licensed software, as is the case with the GPL. Thus, an open source application program (e.g. Oracle) running, like free software, on the Linux operating system may be protected by copyright if its developer /owner requires or wants such protection; unlike a 'Copyleft' license, those addition do not need to be shared with others. (However, Linux, by itself, does not have the features of proprietary software.) As well, a great deal of free software, such as Apache, runs on Windows and there is plenty of free software that operates on a somewhat different licence than GPL.

3. THE LEGAL BACKGROUND: INTELLECTUAL PROPERTY AND COMPUTER SOFTWARE

A range of intellectual property regimes are implicated in the protection of computer software; these state-created legal rights also restrict the uses, domestically and international, of software. This report focuses on the three most important intellectual property rights vis à vis software, namely copyright[7], patent[8], and trade secrets.[9] (Trade mark and trade dress law may also protect software, but they are not explored here.) This section of the report provides a very brief overview of the history of such protection, especially copyright and patent law, and the parameters of that protection. (Some key software legal issues, such as copyright in user interfaces and the requirements for patentability, have been avoided, as they are not as germane to the main thrust of this report.) The emphasis here will primarily be on the development of the US and international legal regimes. The US is not only the largest producer and exporter of computer software, but its jurisprudence on software questions and disputes is the most influential of any in the world; hence an appreciation of even the basics of the international legal regime requires an understanding of US law.

In trying to come to grips with the topic of computer software and the legal domestic and international legal regimes which protects it, we should focus on four key issues as starting points:

- the use and ownership of computer software is essentially a question of power relations between persons, today and for many years into the future;
- the forms of legal protection provided are both contingent and far from coherent in many dimensions; in other words, such forms of protection are not inevitable and could be changed, indeed quite dramatically changed, through reforming the forms and content of legal regulation;
- the granting of intellectual property to computer software is a form of legal subsidization to a particular industry and technology;
- the intellectual property regimes that protect computer software have had a direct impact on the ownership and user regimes that have been established; the alternatives to proprietary software, open source and free software, have been a philosophical and practical response to the existing legal regimes.

These starting points require a bit of explanation.

- Although intellectual property has certain important differences with other types of property, such as land, a house, an automobile or your toothbrush,[10] it is a form of property nevertheless and the law gives the owner of a copyright, a patent or a trademark and other types of intellectual property certain rights over the use of that property. What essentially occurs is that the state creates a property right, for example, it grants copyright in a book, a song or a computer program and, at the same time, it grants sovereignty over that property to an individual, corporate entity or some other group.(Cohen) In granting that right, the state not only give the owner rights (and hence power) over that intangible piece of property, but it also gives the owner of that property certain forms of power over people. On this point, intellectual property has important similarities with property such as land or a house. For example, a mortgage lender is not only given power over a house as a tangible or physical piece of property, but also acquires power over the owner or occupants of that house. Non-payment of the mortgage gives the mortgage lender the power to evict the owner and re-possess that house. Similarly, the owner of a software program, in determining which uses of that software are permitted and which are not, exercises power over not only the software itself, but also over people who may wish to use that software. Moreover, property not only represents power and financial relations in the present, but also determines "what men [and women] shall acquire"[11] in coming years and decades. By becoming signatories to international agreements and treaties protecting software, such as the TRIPS Agreement, the Berne Convention, and the WIPO Copyright Treaty, countries of the South are agreeing to protect copyright in a computer program until, at a minimum, 50 years after the 'author' (i.e. software writer) of a program dies. In other words, copyright —in this case in computer software — not only represents the exercise of power *in the present circumstance*, it also represents power relations far into the future and will truly monumental effects on *their economic futures for decades*, as well as their future use of software. In coming decades, it will become the 'dead hand of the past' controlling their futures.

- Non-lawyers, as well as lawyers, need to keep remembering that laws are created by people and governments and that what is created can also be changed. When we look back critically over the approximately 40 year legal history of the protection of software, we can start to appreciate that this history could, in fact, have been very different. For example in the 1960's, US President Lyndon Johnson established the President's Commission on the Patent System to examine, among other matters, whether computer software programs should be protected by a patent. In its final report, issued in 1966, this Commission specifically and strongly rejected this proposal, detailing a number of practical problems with the possible patenting of software and noting "the creation of programs has undergone substantial and satisfactory growth in the absence of patent protection and that copyright protection for programs is presently available."(Merges) (With this conclusion, the Commission was also expressing the traditional reluctance of intellectual property doctrine to protect the same work or intellectual creation with more than one form of intellectual property protection.) Yet, before too many years had passed, US law began to protect computer programs by patents. The history of copyright protection of computer programs has its own inconsistencies and logical blind spots. (see below)

- The decision taken to establish intellectual property protection (domestically and internationally) to computer software represents, among other things, an extremely valuable legal subsidisation of the multi-national software industry. As was explained above, the provisions of TRIPS and the laws of individual states (and the resulting penalties for infringement) determine the distribution and allocation of current and future wealth, nationally and internationally, as well as access or non-access to computer technology. (For the more than 150 members of the Berne Convention, once protection is granted a work in one country, that work is automatically protected within the borders of all other signatory countries.[12]) The decision to enact copyright and patent laws thus is a decision made by the state to create an artificial scarcity in a given product; such a scarcity then allows the rights holder to operate in a very different fashion than if there was not such a scarcity. There may, of course, be policy-driven reasons that justify the subsidisation that is a consequence of this artificial scarcity; for example, some level of intellectual property protection may be required as an incentive to encourage intellectual creations, such as the writing of new software. Yet, the degree (or necessity) of incentive required is often overstated; the software written by the free software movement was not motivated by the rewards which traditional intellectual property systems provide. So it should not be forgotten that intellectual property laws and treaties represent protectionism and a market/wealth creating intervention by the state.

3.1 Copyright Protection and Restriction

Beginning in the early 1980's, a number of governments in the developed world decided, after extensive lobbying by some (though not all) sections of the software industry, that computer software was analogous to the traditional copyright category of an 'original literary work of authorship' and hence should be protected as a literary copyright. The essential argument was (and still is) that the thousands or even millions of lines of binary code found in a program — the series of instructions (that is, the symbols 'O' and '1' found in infinite patterns in an object code) which are given to the computer hardware — can be best understood, as a matter of legal classification, as forming a literary work. Some will find this difficult to accept. The choice of copyright law as the principal 'home' for computer programs was incoherent for another reason. Traditionally, utilitarian works (that is, functional works that do something or cause another part or piece to do something) such as computer programs have been protected by either trade secret or patent law; previously, copyright law had been employed to protect expressive works, such as artistic and literary works.(Merges) Moreover, Section 102 (b) of the US Copyright Act states that copyright cannot protect "any idea, procedure, process, system, method of operation, concept, principle, or discovery." One could certainly argue that the 'literariness' and function of Java script is rather different from that of a Salman Rushdie novel, yet both are protected under the same copyright regime.

The national copyright legislation in a number of developed countries, such as Japan, the US and across Europe, was amended in the 1980s to explicitly put computer software under the copyright umbrella as a literary work. Acting on the recommendations contained in the 1978 report of the National Commission on New Technological Uses of Copyrighted Works (CONTU), a 1980 amendment to the US Copyright Act, Section 117, expressly recognised the copyrightability of computer programs. Regionally, various treaties and directives (North American Free Trade Agreement, various EC directives) did the same thing over the next 15 years.

Large multinational software companies spent extensive time and resources lobbying for the creation of similar standards and approaches in international copyright agreements; their efforts also extended to countries of the South. A 1994 US law review article explained that:

"[T]he United States government devoted substantial effort over the past decade to browbeating most of the developed world into following its path. Neither the US government nor the many entities desiring uniform protection for their products across national borders are interested in starting a new fight." (Menell)

Both the 1995 TRIPS agreement (Art. 10 (1)) and the 1996 WIPO Copyright Treaty (Art. 5) state that computer programs, both in source and object code, must be protected by copyright. Although TRIPS (Art. 66 (1)) states that least developed countries will not be required to apply this section (and many other sections of TRIPS) until 2006, this deadline is fast approaching and, in the end, they and all other WTO members will have no alternative but to protect computer software under their own national copyright laws. In any event, the WIPO Copyright Treaty provides no 'transitional arrangements' for least developed countries.

As is mentioned in Section 2, copyright law also has important ramifications for FLOSS; putting free and open source software under this legal category means that software licences can be enforced and various conditions can be granted for the use and adaptation of that software.

3.2 Patent Protection and Restriction

During the infancy of the computer era, an era that led, significantly, to a number of important software innovations, software was not protected by patent law in the US or elsewhere. As one US text explains, "[w]ith the growth of computer technology came an early crop of patent applications. In the 1950s and early 1960s, the [US] Patent Office met these with a uniform response: whatever software is, it is definitely *not* patentable subject matter."(Merges) In the early 1970s, US software patent cases examined whether an 'invention' based on the use of a mathematical algorithm could be 'patentable subject matter', to use the legal term of art. Favourably citing the already mentioned 1966 Report of the [US] President's Commission on the Patent System, which opposed the patenting of software as noted above, a 1972 US Supreme Court case, *Gottschalk v. Benson* (409 US 63) held that the computer algorithm patent could not be granted a patent. However, nine years later in the case of *Diamond v Diehr* (450 US 175 (1981), that same court upheld a software patent. In succeeding US cases, there were continuing debates on the patentability of software and the creation of new tests to determine whether particular types of software were patentable. A more recent US case, *State Street Bank & Trust Co. v. Signature Financial Group, Inc.* (149 F.3d 1368 (Fed. Cir. 1998)) further expanded the ambit of software patents by allowing the patenting of what are known as 'business method patents'.

A limited number of other countries, such as Japan, also permit the patenting of software. In Europe, standalone software patents — or programs 'as such' — are not permitted by statute (see e.g. Article 1 (2) (c) of the United Kingdom Patents Act, 1977), but software which is an integral or functional part of some other machinery or invention can be patented; the skillful drafting of software patent applications can often bring such patents into the requisite 'integral' category. For the last several years in Europe, there has been a sharp debate, chiefly pitting large multinational software companies against small and medium-sized software developers, as to whether the US and Japanese approach to software patents should also be adopted in Europe.

Internationally, there is no explicit wording that signatory members of the TRIPS Agreement must allow software patents, although this wording and its meaning is not straightforward; see section 5.4 for a fuller discussion of the TRIPS requirements. Unlike the legal situation with copyright, the fact that a particular software program is protected by a patent in one country, such as the United States, does not give that program patent protection in another country unless there has been a specific patent granted for that patent; most countries in the South do not allow software patents, although this situation is beginning to change.

3.3 Trade Secret Protection and Restriction

Trade secret law operates on a different basis than the copyright and patent protection offered to software. One commentator outlines the US approach as articulated by its Uniform Trade Secrets Act, which all but a handful of US states have ratified and enforce.

"Rather than focusing solely on expression or demanding novelty as a prerequisite to protection, the law of trade secrets will protect the ideas underlying particular software — including the software's structure or architecture and organization, and various features, routines and processes within the software, novel or not — so long as those ideas are not generally known (or readily ascertainable from the marketed software) and give, or have the potential to give, a competitive advantage by virtue of the fact that others do not know them." (Cundiff)

Given that access or non-access to the source code is such a key computing issue and that most proprietary software owners make great efforts to protect such code as confidential (non-disclosed) information, the relevance of trade secret law is immediately obvious. For even if a software developer does get access to the object code of a program — typically by taking out a licence, — such access seldom permits access to the source code and, further, access to the object code does not compromise the trade secret status of the source code.

"While it may be physically possible, and legally permissible, (footnote omitted) to use devices called 'disassemblers' or 'decompilers' to get a printout of some or even many portions of the object code, it is generally extremely difficult, if not completely impossible, even for one skilled in computer science to recreate the source code or fully understand the software based solely upon an examination of the object code. This is so in part because while the object code can show what happens at what point in a program, it does not explain why; further, it does not explain what information is being stored in or generated by each portion of the program, making it difficult, if not impossible for one reviewing the software to determine what data the program is analyzing or working with at each step along the way." (Cundiff)

Computer service and maintenance manuals, which may be of assistance in the reverse engineering of software, may also be protected as trade secrets.

In the past decade, trade secret protection for software has taken on an international dimension as well, including for countries of the South. The TRIPS agreement is the first multilateral agreement that deals with trade secrets in any level of complexity as previous multilateral conventions generally ignored or avoided the issue.(Gervais) Article 39 of TRIPS sets down the national requirements of signatories and establishes that all such countries must enact trade secret legislation which would, if requirements such as those regarding the confidentiality were met, also cover software. The fact that computer source code may be a trade secret has a number of implications for countries of the South, especially the question of technology transfer that is examined in Section 5.4.

As its very name implies, the codes in FLOSS programs are not protected under trade secret laws as they use an open, non-confidential code, such as Linux.

In conclusion, the three regimes of copyright law, patent law and trade secret law provide, individually and collectively, a formidable legal fortress for computer software, both domestically and internationally.

4.THE GROWTH OF FLOSS ACROSS COUNTRIES OF THE SOUTH[13]

This section of the report attempts to give a 'snapshot' view of some of the more important and typical (which often means small scale) developments in FLOSS across ten countries of the South; at different moments, conflicts with proprietary systems and uncertain futures or failures are mentioned. Rather than attempting to be encyclopaedic or all-sided, the compilation effort, done here on a country-by-country, has attempted to focus on the breadth of new software projects, key innovations, and unique attempts to stem the so-called digital divide. One reading of this section might suggest that the future is rosy for alternatives to proprietary systems and that the road ahead is straightforward for FLOSS; in fact, neither view is correct and there are critical conflicts in the current conjuncture and many barriers to face in the years ahead. (These problems are examined in other sections of the report, particularly sections 5 and 6). Although the conclusion that "the developing world is leading the developed world in open source adoption"(Weerawarana and Weeratunga) is perhaps somewhat overstated, it is true that the entire globe may be able to learn some valuable and transferable lessons from the range of FLOSS projects already established or about to come on stream in the months and years ahead in the South. The software world, especially for open source and free software, is a fast-moving field; most of the developments noted here have occurred over the past two years and there may be other important changes in the FLOSS and proprietary software picture in the months ahead.[14]

China

China is becoming an increasingly important player in the world software scene. The lead author of a July 2003 US Rand Corporation report on global technology specifically exempted China from his dismissive comments that most countries competing with the US for software business were "losers or laggards". A month earlier, a senior World Bank official predicted that Linux would "become the No. 1 operating system in China and India soon." In 2003, the Chinese government made further announcements of plans to develop its own domestic software industry based upon the Linux operating platform and more are expected in 2004.

In February 2003, Red Flag Software Co. Ltd., a Chinese open source company founded in 1999, won its largest contract to date; it will be installing Linux operating systems and related equipment in 77 postal districts across China. Earlier Red Flag won open source contracts with the Beijing municipal government. In 2001, more than one million Linux suites were installed through personal computing vendors in China.

India

During a May 2003 visit of Microsoft chairman Bill Gates, Indian President Abdul Kalam said that "open source code software will have to come [to India] and stay in a big way for the benefit of our one billion people," adding that such software provides developing countries with the best opportunity to modernise their computing infrastructure.

Commentators say Linux is starting to challenge the continuing dominance of Microsoft's Windows — much of the latter being 'pirated' (See Section 5.6) — and India appears to be a key battleground between the two competing software visions. Microsoft is showing no signs of conceding defeat. Late in November 2002, Microsoft chairman Bill Gates announced that his company would be spending more than $US 400 million in India over the next three years; a significant percentage will be spent on training 3.5 million students and 80,000 school teachers in the use of Microsoft software in more than 2,000 school labs. This will lead to further computer software 'lock-in' in India. (See sections 5.3 and 5.4 for more on this phenomenon.)

Some novel FLOSS projects are being developed in India. Simputer is a pocket-sized computer developed several years ago by four Indian computer technologists that allows online access, basic word processing, text-to-speech capabilities, and operation in four Indian languages. The projected cost is less than $US 200 which necessarily means it is using open source rather than proprietary software. Although the word Simputer stands for simple computer, MIT sociologist of science Kenneth Kensington says, "I don't know of anyone else in the world who is producing a comparable computer at this price."(Stikeman) A prototype Simputer has already been built and commercial licenses are being negotiated. Certainly Simputer will not replace a desktop PC, but members of the non-profit development trust behind this new "Third World appropriate" technology expect it will become popular in rural India, an area essentially cut off from the new information age, and will be purchased by neighbours banding together to buy the machines for communal use. "We are quite used to sharing here," explained one member of the trust.

Thailand

A 2001 survey by a leading U.S.-based computer survey firm found that more than 25 per cent of Thai companies used the Linux operating system.

In June 2003, Microsoft made huge price reductions in the cost of their latest software, Windows XP and Office Suite in Thailand; the deal, which is only for Thai consumers, is intended "to curb the spread of the Linux operating system" in Thailand, say commentators. The software combination now retails in Thailand for only $US 35.80 — less than 10 per cent of the usual price elsewhere — while a desktop computer loaded with these two software packages is $US 298, excluding taxes. In other parts of the world, according to news reports, a standard edition of the Office XP software alone costs $US 399.

The initial group of computers, being purchased in 2003 as part of a government-subsidised programmed to increase computer usage among the poor, were loaded with the Thai-language version of the Linux operating system.

South Africa

In February 2003, a leading South African government council recommended the government adopts an official policy that promotes the use of open-source software — but would stop short of jettisoning proprietary applications. "The primary criteria for selecting software solutions will remain the improvement of efficiency, effectiveness and economy of service delivery by (the) government to its citizens," reads the policy proposal. "OSS offers significant indirect advantages. Where the direct advantages and disadvantages of OSS and PS (proprietary software) are equally strong, and where circumstances in the specific situation do not render it inappropriate, opting for OSS will be preferable." The government is the largest single buyer of computer technology in South Africa and, "when a country or community is at the mercy of a technology provider, and powerless to determine or shape its own fate, that the situation becomes problematic and an intervention is required," said the chief executive of the Council for Scientific and Industrial Research.

Existing open source systems, such as those operating the computers within the Department of Health in the North Cape have been much more reliable than systems in other parts of the government service using proprietary software, the South African minister of public health and administration said in May 2003.

In 2002, Microsoft promised to donate more than 30,000 software licenses to schools in South Africa.

Uganda

At Uganda Martyrs University in Kampala, Uganda, two FLOSS projects were initiated in 2002-03 at this small university. The first and also the most ambitious one, aimed at replacing all proprietary software at the university campus with open source software. In the second project they aim to revive old 386 computers in order to distribute them to local schools. "Old computers are not, as most of the people in the industry think, useless and they are available in plenty. Mechanically they are still working in most of the cases, but they are outclassed by the software industry. New software requires too much capacity of the computer and the older software that is needed to get some performance is no longer available," a university computing expert explained. The ICT team of the university, together with the University of Nijmegen in the Netherlands and a small German consulting firm, are trying to implement a lightweight Linux version. When successful, a new standard distribution for primary and secondary schools can be prepared that will be used on the donated computers from the university.

Namibia

In November 2002, SchoolNet Namibia, a mostly volunteer organisation providing computing resources to this poor southern Africa nation, publicly rejected Microsoft's offer to put the Windows operating system in its schools. Instead, they decided to keep their existing open-source Linux systems. What had initially appeared to be a helpful and charitable donation to this nation's educational computing resources turned out to be otherwise. In order to obtain fifty inexpensive laptop computers and copies of the Office Pro application program, it turned out that SchoolNet would have to pay about $US 4,500 per school for the cost of using Microsoft's operating system in what had been billed as a Microsoft 'pilot project'. That price was simply not affordable by SchoolNet and the donation was refused.

Lebanon

One example shows how difficult it is for developing countries to challenge or even slightly modify the powers of closed source proprietary software monopolies. In Lebanon, there was vigorous opposition in its Parliament during both 1997 and 1999 to draft government legislation on the subject of computer software. A number of MPs argued that software should not be protected by copyright and, in particular, that computer copyright owners, such as Microsoft, should be required to grant a compulsory software licence to poorer students and to educational institutions in Lebanon. As a result of pressures applied by Microsoft, Adobe, and other software multinationals, Lebanon was put on a US Trade Representative Special 301 Watch List (that is, given a warning that the US could decide to impose trade sanctions) for considering such a reform. In the end, Lebanon was forced to comply.

Mexico

On occasion, major new OSS projects and programs have been announced with great fanfare, but then have failed to take off. What happened in Mexico provides an object lessons for other countries of the South. In 2001, Mexico's Scholar Net project, which had planned to install open source software in 140,000 schools, was announced. What was particularly noteworthy was that these installations were slated to save more than $US 100 million compared to the projected costs using Microsoft products and various reports hailed it was an important breakthrough.(see e.g. White, Story) But two years later in 2003 only a small number of schools had actually installed open source software and Windows remained the pre-dominant operating system in Mexican schools. Observers suggest that the paucity of trained technicians in Mexico, the lack of effective support from the Mexican Secretary of State for Public Education, and the short cuts taken were the main reasons for the failure of this project to take off. "The assumption was made that to implant free software in schools it would be enough to drop their software budget and send them a CD ROM with GNU/Linux instead. Of course this failed, and it couldn't be otherwise, just as school laboratories fail when they use proprietary software and have no budget for implementation and maintenance."(Villanueva)

Brazil

Brazil, South America's most populous countries, is one of the earliest centres of free software development in the South and the Lula government is now formulating a number of programs to facilitate the spread of FLOSS in that country. Four Brazilian cities (Amporo, Solonopole, Ribeirao Pares, and Recife) have passed laws giving preference to or requiring the use of OSS (including from service suppliers).

Peru

A bill presented in November 2001 to the Peruvian Congress advocating much wider use of free software by the government sparked a great deal of debate and controversy. Bill 1609, entitled "Free Software in Public Administration" and proposed by Congressman Edgar Villanueva, was aimed at trying to ensure Peru finds "a suitable place in the global technological context." (Villaneuva) The contents of the bill were strongly attacked by Microsoft, but Villlaneuva's lengthy 8 April 2002 open letter in response received wide circulation and became a rallying cry for FLOSS activists across South America and the South more widely. Villanueva says that a growing number of young people in Peru are getting behind the bill and free software and many have offered to hold marches in support. "It is the youth that needs to drive its creativity, its intelligence, its intellect … there are many people who can create their own employment through [the use] of free software."

5. CRITICAL ISSUES

Introduction

Building on the technical and legal background provided in sections 2 and 3 and the news 'highlights' contained in section 4, we are now ready to examine a range of critical issues that are implicated by the computer software debate for countries of the South. Two inter-related areas demand particular study and comment:

- the affordability, use, and accessibility of the competing types of software;
- the relationship between the various types of software and wider economic and social development for countries of the South.

In this report, these two over-arching issues are examined under six sub-topics:

1. Software Costs and Licensing
2. The Hardware / Software Interface and the Question of Hardware Replacement/Updating
3. Software, Technology Transfer, and Technological Independence
4. Software patents and innovation.
5. Software employment and training issues
6. Unauthorised software use (software 'piracy')

Under each of these six sub-topics, this section of the report compares and contrasts a number of factors involving the current and possible future use of proprietary software and FLOSS in the South.

5.1 Software Costs and Licensing

This is the first part of Section Five — and also one of its longest. On one level, this makes logical sense. The costs, primarily the licensing costs as well as the follow-up maintenance costs, of proprietary software vs. FLOSS are one of their key distinguishing features and, as this report suggests, one of the main advantages of the latter is, in fact, its significantly lower cost, both as an initial licence item as well as the maintenance and servicing costs over the life of the software. In the South, where per capita incomes are far lower than in North America and Europe and where budgets for capital expenditures (e.g. for the equipping of government offices in an entire province or in several school districts with computers for the first time) are likely to be far more stretched, the question of the cost of the software is obviously an important matter. Yet, on other level, other questions, such as the adaptability and flexibility of the different types of software and the employment possibilities each offer to the South and the broader questions of technology diffusion vs. technology transfer are, in fact, more important than the actual cost of a single piece of software. Indeed, an over-emphasis on the most obvious or observable cost issues in poorer countries may lead one to the view that the preferred global software 'solution' is differential pricing of proprietary software as has been proposed in the case of anti AIDS/HIV drugs. (This approach and this proposed 'solution' is encouraged by the supposedly philanthropic or charitable donations of proprietary software to countries the South and is examined below, as well as in Section 6.9) One relationship that is integrally linked to the high cost of proprietary software is that which exists with unauthorised use of software, also known as software 'piracy'; this question is examined in Section 5.6 (see also Ghosh).

Only a few years ago and before there were widely known competitors with or alternatives to proprietary software, such closed source software was already expensive for countries of the South. But if a government office, school, or health clinic wanted to operate a computer system, there was really no other alternative than the use of proprietary software — though, in truth, much of the software had been copied without the permission of the rights holder, it was 'pirated'. While countries of the South should not, of course, be expected to rely on used software donated by users in industrialised countries, the change in software practices undertaken by the London-based charity, Computer Aid International, gives us an insight into the new software world that is now opening up. (Computer Aid collects used personal computers (i.e. still operational computers that are surplus to the requirements of UK businesses, groups and individuals) and ships them to governments and groups in the South that need them; see more in Section 5.2) Most of the machines it receives come 'loaded' with proprietary software, particularly Windows, and, until quite recently, Computer Aid technicians re-loaded these computers with Windows and a software package such as Word and sent them across the globe with the appropriate software licences to accompany them. But that practice has now ceased as Tony Roberts, director of Computer Aid, explains:

"Until recently, there really was no other choice…but now there is. Instead of using proprietary software, we now re-load all of the machines with open source software; it is much cheaper for us and the end-users and is a much appropriate technology for use in places such as Africa…. With the exception of a few parts of South Africa, there is not a single government or a school system anywhere in Africa (emphasis added) that can afford the costs of a Microsoft licence for their school systems." (Roberts interview in Story)

A growing list of governments and groups in the South had already reached the same conclusion and, as highlighted in Section 4, more are doing so on almost a monthly basis: the prices for proprietary software and the requisite licences (e.g. for schools) are beyond the reach of all but a small elite in the least developed countries and beyond the reach of most groups in all countries of the South.

Here are some specifics of this problem, illustrated by evidence taken from interviews with university officials in the South over the past few years. Take the costs of Microsoft's licences that such institutions must purchase to operate their hardware configurations. Microsoft generally follows a practice of charging the same price for its software products around the world *without regard to widely disparate average income levels* (Reuters, 16 October 2001). The same is true with regard to educational software licenses; Microsoft licensing officials in Vietnam and Ecuador have confirmed that the "per seat licensing fee" for universities in those two countries is essentially the same as Microsoft charges Harvard or Oxford University. (In the same vein — and setting aside, for the moment, the question of proprietary software 'donations' — an elementary school in Soweto, as a licensee, is treated in same way as is a school in a suburb of Boston, US.) Although the prices are today rather higher, the Windows (operating system) and Office (word processing application software) licences for 100 PC's at a university anywhere in the world cost $US 5,500 in the summer of 2002. In fact, numbers of persons that I interviewed who checked the local price for software/hardware packages in countries such as South Africa and Argentina found that prices were even higher than in the US (e-mail correspondence on file); part of this price differential can be explained by higher hardware costs, which included pre-loaded software, in these less competitive markets. By comparison, the basic Linux software system can be downloaded from the Internet for free, though there will likely be follow-up and servicing/repair charges. But proprietary closed source systems also require and encourage service contracts. Most importantly, individual licensing charges are not levied for use by each individual computer; FLOSS software is software that is shared.

But the situation regarding the high cost of proprietary software is actually far worse than this; it can be highly misleading to compare the price of software globally without looking at the ability of potential purchasers to buy that software. One study, which compares the license fees for a standard type of proprietary software to per capita income in 176 countries, was completed in late 2003.(Ghosh[15]) Its focus is on 'effective' purchasing power; the study calculates how many months of wages it would take an average person in each country to purchase a typical Microsoft operating system and word processing application system. The example chosen is Windows XP, plus Office XP which, according to an Amazon.com price list of June 2003, cost $US 560 in the US. Here are a few of the 176 countries surveyed, five from the South and two industrialised countries. (*See Table 1*)

And there are even more startling statistics. In the Democratic Republic of the Congo, the effective cost of this software package would be $US 199,394 and it would take the average wage earner 67.83 months — more than five years — of earnings to purchase Windows XP and Office XP. Overall, the Ghosh study and its calculations clearly reveal the actual burden of proprietary software prices for poorer countries.

Looking again at universities, here is one brief case study of the costs that were involved in the licensing of software at the Uganda Martyrs University Nkozi (UMU), a small university (362 full-time students; 632 part-time students; 84 faculty and administrative staff) located in Kampala Uganda.[16] The UMU has a total of 135 computer workstations which formerly used two servers running

Table 1: Effective Cost of Windows XP

Country	Per Capita GDP [a]	'Effective' Cost of Windows XP [b]	Equivalent no. Months of Average Income [c]
Bangladesh	350	56,401	19.19
Brazil	2,915	6,777	2.31
Cameroon	559	35,319	12.01
Trinidad and Tobago	6,752	2,926	1.00
Vietnam	411	48,011	16.33
United Kingdom	24,219	816	0.28
United States	35,277	560	0.19

a: The per capita GDP (gross domestic product), i.e. the average individual income in US$, for that country (based on the 2001 World Bank Development Indicators Database);
b: The "effective" cost of Windows XP, that is, the cost as reflected by the per capita GDP in that country;
c: The number of months of average income which it would take to purchase Windows XP operating system and application program in that country.

Microsoft software. Microsoft charged UMU a total of $US 10,997 per server per year for use of the required licensed software (ISA server enterprise edition; Microsoft Windows 2000 server; N2H2 Academic Licence). Hence, the annual licensing fees UMU paid to Microsoft totalled $US 22,000. But the computing staff recently concluded that two servers were not sufficient to handle UMU's growing computer traffic and proposed purchasing two additional servers which, in turn, would require additional per server licences from Microsoft. (To give a context for this decision, the University of Kent, a medium-sized British university, operates about 60 computer servers.) But when they received a quotation, computing staff decided that the University simply could not afford the additional servers. "I started to get really scared that it was going to cost far too much money and be too expensive," a member of the UMU's computing staff, explained in an interview in early August 2003. Instead, UMU has decided to switch to open source software. The "migration is still not over", the UMU computer expert said, but he said he expected significant cost reductions in the software costs in coming years. "Here in Uganda, we cannot afford to pay the same licensing rates as they do at an American or British university," another UMU staff member explained.

Sometimes it can be difficult to make direct costs comparisons as commercial intelligence considerations come into play; in such cases we must rely on estimates. When the Korean government agreed to purchase 120,000 copies of a Linux-based from a local company in 2002, industry analysts estimated that the government would pay about one-fifth of the value of equivalent Microsoft software.(Shankland)

Microsoft and other proprietary software firms counter this orientation. While agreeing that the initial price of open source software is obviously far below the cost of its own proprietary software, Microsoft says its products more than match the 'total costs of ownership' (TCO) of open source software. (The TCO includes, in addition to the initial licence fees, the costs of installation, ongoing maintenance costs and repairs.) However, in a November 2002 survey that Microsoft itself conducted of software developers and other information technology workers in the US, Brazil, France, Germany, Sweden and Japan, a leaked memo reported that "overall, respondents felt the most compelling reasons to support OSS was that it 'offers a low total cost of ownership'". (Ard) In any event, the TCO cost needs to be calculated rather differently in countries of the South. Most TCO studies have been conducted in richer industrialised countries where labour costs are relatively much higher.

"When labour costs are high, labour-intensive components of the total cost (such as support, customisation, and integration — i.e. everything other than the software licence fee, communication and hardware costs) represent a high share of the total cost, making the licence fee itself (which is not present in the case of open source software) less crucial. In contrast, when labour costs are low [as in most countries of the South], the share of the licence fee in the total cost of ownership is much more significant, even prohibitively so."(Ghosh)

Another complicating issue involves the 'deep discounting' of proprietary software and donated (that is, given at no initial cost) software. One computing expert in Argentina has commented that:

"A common marketing practice for many proprietary software vendors is to deep-discount their prices for one-time sales. This encourages the customer to acquire the licenses, which in turn helps establish widespread use of their software. Once the customer is using this particular piece of software, prices revert to normal for further purchases, which means any new compatible products he may need to expand his business have now a much higher price, and by that time he has no other choice than to purchase from the same company, because it's the only one that can provide compatibility with his installed base." (Heinz)

Essentially, the same issue arises with donated software and, increasingly, some countries are becoming sceptical about the actual motivations of the gift giving and worried about the technological 'lock-in' that may result. (See Namibia, Section 4) On other occasions, software has been donated, such as in South Africa, just as that country appeared likely to embrace FLOSS more widely. Microsoft, as part of its proposed settlement with a number of US states following its recent US anti-trust prosecution, suggested that it donate tens of thousands of free software licences to school and low-income communities located in those states. Yet, some states rejected this offer and the donations after calculating that, over the long-term, the licensing costs would be substantial. The donations are not dissimilar, because the computing needs in the South are so pronounced, turning them down is not an east decision to make.

Another complaint by educators and IT specialists in the South concerns the standardised approach to licensing. Although the situations are widely disparate, the proprietary software licences made available in the South are, in most cases, exactly the same as the licences offered, for example, in the US or the UK; the same 'one size fits all' restrictions apply no matter what are particular computing requirements, needs, and financial capacities of the end user, whether she/he is located in Accra or Atlanta. As a Ghanaian software developer commented, "the market here is too small for locale-specific versions of software and consequently. We have UK/US versions resold here. As is."(Sohne, in Story) The problem does not arise only with proprietary operating system software. A professor at the School of Architecture at the University of Natal in South Africa has explained that, because of licensing restrictions, most of the schools poorer students, particularly from Zimbabwe, were not able to afford the purchase of home copies of the specialised 3-D modelling software needed for architectural design; the software licence restricts use to 'at school' use.(Wang interview) Such global market strategies and business models, let alone the underlying philosophy, can hardly meet the urgent computing needs of countries of the South.

In countries of the South, the cost savings are a particularly strong reason that is given for the growing popularity of non-proprietary systems. Studies have shown that the high prices of software were one of the main barriers to the adoption of computer systems by local governments, hospital and health care facilities, and by numerous other organisations across the South. In richer countries, by comparison, the costs of software are sometimes a 'forgotten' item in the budgets of companies, organisation and governments. But a recent study underlines the importance of software costs in the South, especially in the poorest countries. "A software licence that costs say £500 is not a great barrier for most UK companies; it is worth paying to save a few days (or even hours) of employee time. In the developing world, this is not true and free desktop software looks much more attractive."(Peeling)

Yet a *strong word* of caution is advised; an over-reliance on the potential savings to be achieved by the use of non-proprietary software sometimes can blind organisers to unforeseen and ancillary costs. As explained above in Section 4, a Scholar Net project was planned in 2001 and 2002 for Mexico which involved the installation of computer operating in 140,000 school computer laboratories. Which software to choose for this large project? Organisers estimated it would have cost a minimum of $US 885 to install Windows 98, Microsoft Office, and a server running Windows NT in each of the schools. The software costs thus would have totalled $US 124 million. But when it was possible for only $US 50 to buy a single set of installation CDs and a manual for RedHat Linux (the license allows the software to be duplicated and reinstalled without limits) and there were only small initial installation costs, Mexico Scholar Net chose Linux.(White) Yet, this open source project has been all-but abandoned (as explained above) because all of the ancillary costs of both installing and operating this software over the longer-term had not been properly taken into account and the required education and ongoing support programs were overtaxed. Like most other forms of technology, FLOSS is *not* a mere consumption item and requires far more expenditure and resource allocation than a 'one-time only' payment, however low that is.

5.2 The Hardware/Software Interface and the Question of Hardware Replacement/Updating

In industrialised countries, an ethos — and indeed an expectation — has been created which suggests that the upgrading of computer hardware needs to be carried out every few years. Advertising for 'state of the art' computers powered by 'lightning-fast' processors is commonplace; many view them as 'must have' purchases. The merits and costs of frequent hardware upgrades or, more commonly, complete hardware replacements, in industrialised countries is obviously beyond the scope of this report. But for countries of the South, the issue must be approached very differently because, in most cases, the burning question is how to obtain *a first computer*, not an upgrade or a replacement; additionally, the much lower income and expenditure levels frame the hardware acquisition debate very differently. This brief section looks at the software / hardware interface and the role of proprietary software vs. FLOSS, as well as the question of what to do with the burgeoning piles of still-operable computers that are piling up in the landfills of Europe and North America. It is an issue that is often overlooked in the current software debate.

Although it could be viewed as a proverbial 'chicken and egg' question — which came first? increasingly sophisticated hardware? or increasingly sophisticated software that requires increasingly sophisticated hardware for its operation? — what cannot be disputed is that the almost constantly updated proprietary software programs and packages that are created require the regular upgrade or replacement of hardware, often every few years. Take the Windows XP/.Net operating system that was introduced in 2001 by Microsoft. Writing for a US audience, a number of commentators and detailed technical studies have revealed that this operating system would mean that:

- because of new licensing restrictions, users will be required to purchase separate XP software for each PC they own;
- the use of Windows XP will require 265 megabytes of hardware memory, an uncommon amount on computers purchased more than a year before the release of Windows XP.

As a *Business Week* magazine computer expert noted in late 2001 "Windows XP …will place a lot more demands on your computer, so millions of people, especially with those more than two years old, may need new ones."(Wildstrom) New application programs created by other proprietary software companies often will simply not work — or work as intended — on what some label 'outdated hardware'. Whatever the merits of these new proprietary operating systems and application programs, it is beyond debate that most governments, educational institutions and other organisations in the South, as well as individual computer users, simply cannot afford yearly or bi-annual new hardware purchases. To divert limited funds into the replacement of existing hardware for a limited few means that many others will not be able to obtain any computer whatsoever. (Or it would divert funds into new information technology purchases that could better be spent on other economic and social priorities.) By comparison, operable hardware that is five or even ten years old (and hence has lower and less sophisticated technical capabilities) is often perfectly capable of running most free and open source programs. The ability to get off the hardware replacement / upgrade 'treadmill' is a key financial advantage of switching to FLOSS for countries of the South.

This replacement 'treadmill', which shows no signs of abating in industrialised countries and is likely even speeding up, has led to a situation in Europe and North America in which more than 90 per cent of still-operable computers are dumped into landfill sites. "Every year in the UK approximately 3 million PCs are decommissioned and are no longer in use. A great many of these un-used PCs are in fine working order." (website of Computer Aid International) Not only is this a growing environmental issue in industrialised countries, but also it is a serious waste of badly needed global computing resources. "In the developing world 99% of schoolchildren graduate from high school not having seen or touched a computer in the classroom." (website of Computer Aid International) And so while a '486' personal computer or even a computer with a Pentium 100 MHz processor may be viewed as obsolete to organisations in Europe or North America, especially if they want to use proprietary software, such computers, properly configured and equipped with the appropriate free or open source software, can make an excellent workstation or Internet access device for schools and other organisations in poor and middle-income countries.

It is the re-cycling and re-use of such hardware that has become the mission of a number of organisations from industrialised country. As is explained in Section 5.1, one example is Computer Aid International, a London-based charity distributing used personal computers to groups and individuals in poorer countries. (There are similar organisations in other industrialised countries.) Since 1998, Computer Aid International has shipped more than 20,000 fully refurbished computers to more than 70 countries; schools are one of the priority recipients. Today, Computer Aid technicians exclusively load open source software on such machines as the charity realises that such software is much more adaptable and affordable for their needs. There are also a growing number of South-to-South hardware re-cycling projects (again involving FLOSS); Uganda Martyrs University in Uganda is one example of university that is currently operating such as a scheme.

5.3 Software, Technology Transfer, and Technological Independence

The transfer of technology remains one of the leading issues on the agenda of the North-South development dialogue; in the information technology / Internet era, computer software has become an especially important item on that agenda. After a brief backgrounder on the issue and mention of the articles in the TRIPS Agreement that deal with this issue, the main part of this section of the report compares proprietary software and FLOSS and their respective capacities to transfer technology.

For more than three decades, the transfer of technology to countries of the South has been a recurring theme in various multilateral discussions focused on economic development, the stimulation of further innovation, foreign direct development, and, more recently since the late 1980s, on the spread of the global intellectual property regimes. This debate, especially its intellectual property component, took on a higher profile during the lengthy GATT discussions of the Uruguay Round that culminated in the TRIPS Agreement of 1995. "Probably the most traditional argument for IPRs protection in developing countries is that technology owners are less willing to transfer proprietary knowledge to countries with weak protection because of the risk of 'piracy'."(Braga and Fink) A corollary or consequence of this logic is that strengthened, indeed globally harmonised, intellectual property protection would result in significantly increased levels of technology transfer to countries of the South. And although some countries of the South expressed certain reservations about this logic — indeed numbers feared that increased intellectual property protection would, instead, limit access to technology and hinder innovation in their countries and regions — they signed on to what has been labelled in some circles as one of the 'TRIPS trade-offs' in the mid-1990s. Why? While there is ongoing academic and economic debate about the precise role of expanded levels of technology in the overall developmental process, there is little debate that such countries possess sub-optimal levels of technology and that this weakness is one of the key barriers to their economic growth and international competitiveness.

In exchange for agreeing to significantly higher levels of intellectual property protection and enhanced enforcement measures, countries of the South successfully argued that TRIPS must contain language mandating the transfer of technology.

Articles 7 and 8 of TRIPS, found in that agreement's 'basic principles' section, as well as Article 66 (2), mandate the transfer of technology, and specifically, in the latter article, to least developed countries. Subsequently in the Doha Ministerial Declaration of 2001, the World Trade Organisation agreed to create a working group that was tasked to examine "the relationship between trade and technology transfer and of possible recommendations on steps that might be taken within the mandate of the WTO to increase the flows of technology to developing countries." The creation of this working group was an agenda demand of such countries because, five years after the signing of the TRIPS Agreement, there had been very little evidence that industrialised countries were acting on their TRIPS commitments to promote and facilitate the transfer of technology. In the Africa Group proposals of 4 October 2001 for an "alternative text to the Draft Doha Declaration", these nations asked that "developed country Members shall put into *immediate effect* meaningful incentives for the purpose of promoting and encouraging technology transfer." Today and more than two years after Doha, there have been no significant improvements in the levels of technology transfer from North to South nor 'meaningful incentives' created to do so and a growing number of countries in the South are again asking: what is the place of technology transfer in the global economy and trading system? Does it remain a matter of only tertiary concern to industrialised countries?

Technology transfer issues unfold with particular clarity in the case of computer software. Moreover, the differences between the two types of software, proprietary and FLOSS are stark.

As an initial matter, the legal acquisition of proprietary software by users in the South (or the North) is generally not conducted as a sale; rather, such software is usually licensed, increasingly on an annual basis and requires, therefore, the payment of annual license fees. (Story) But this legal transfer is not a technology transfer. The definitions of 'technology' and 'technology transfer' contained in the UNCTAD draft International Code on the Technology Transfer "excludes goods that are sold or hired [or leased] from the ambit of 'technology'. Thus it is the knowledge that goes into the creation and provision of the product that constitutes 'technology', not the finished product or service." (UNCTAD, 2001) It is precisely as a finished consumer product that its manufacturers, wholesalers and retailers conceive proprietary software; the licensing of such software is therefore outside the ambit of generally accepted notions of North-to-South technology transfer.

What is particular about the licensing of proprietary software is that the user does not get access to the all-important source code; rather, the source code is completely inaccessible to that user and protected by the twin legal regimes of copyright and trade secret law (and perhaps also by patent law). And the licensors intention is that it be kept that way as a non-accessible secret that is not shareable with others. The licensing terms make this explicit. It is analogous to the sale of a tractor engine by an equipment manufacturer to a Southern farmer, except that the farmer is not allowed, by law, to look inside that engine and understand how it works, change the spark plugs him or herself, improve its performance, modify it to perform tasks that the vendor does not wish it to perform, or, for example, figure out how a cheaper and more efficient version could be manufactured in his or her own country. Even though the licensing of such software is *not* a technology transfer (see 5.3), even if it was, one of the key transfer of technology issues is whether the technology which is transferred is capable of local adaptation. (Roffe and Tesfachew) As they have written "in building technological dynamism, what matters most is not the transfer of technology *per se* but its adaptation and assimilation in the local economy." The source code of proprietary software is not adaptable because, in the first place, intellectual property protections make it unavailable.

By comparison, FLOSS requires that the source be made available — in other words, capable of adaptation and assimilation to local conditions — for all who want or need it (or at least under the terms of the GPL and other similar licences.) Miguel de Icaza, a Mexican-born open source software developer who now is president of Boston-based Ximian, explains that

"The beauty of free software…. is that part of the freedoms you receive is the freedom to learn from other people's techniques, strategies, and focus on problem solving. Something that has been unheard of in this industry (although it is a pretty common thing in science). So people have a chance to join the effort, and be part of the team of people that are producing knowledge, culture and, as a result, wealth." (de Icaza)

It is this unrestricted access to the source code which not only creates the potential for a 'spin off' IT sector to grow in developing and least developed countries, but also allows users and other software developers to create their own software tailored to their own needs and their own national and regional languages.

FLOSS, by permitting access to its source code, allows users or members of a users group to 'de-bug' faulty programmes and builds self-reliance in permitting them to do their own repairs and servicing. (Developing such skills does, of course, require training; see section 5.5). By comparison, proprietary software does not permit the user to make his or her repairs and to attempt to do so may void the guarantee. To return to the tractor analogy, the FLOSS 'tractor' allows the farmer, who may be quite remote from a repair shop, to diagnose faults himself or herself or to call upon nearby farmers to assist in the repair process and allows others to learn from the repairs. This self-help approach, not permitted or feasible with closed source proprietary software, can dramatically cut ongoing computer usage costs, a factor of significance importance for poorer nations. In other words, FLOSS permits, indeed facilitates and encourages, technology transfer. Lacking such adaptability, proprietary software results in mere product diffusion.

The inflexibility and technical 'biases' of proprietary software further lock countries of the South into a pattern of dependency and economic stagnation; FLOSS systems promote technological self-reliance and independence. Again, here is how Miguel de Icaza puts the case for the role of free software in countries of the South.

"I believe that Free Software will help countries with developing economies (like Mexico) to get a competitive advantage that they have lacked for so long. Most of these countries missed the industrial revolution, and for one reasons or another, they depend on external technology to keep up with the times. Free Software helps in the fraction on depending on external technologies. For example, countries with developing economies can now avoid depending on proprietary software: they can keep the money they spent on proprietary software to themselves, and use it to either develop themselves, or they can use that money to produce free software that will solve their problems (and hopefully other countries problems). The case of Mexico is the one I am most familiar with: Mexico does produce very little technology, depends a lot on foreign technology and pretty much our main exports are raw materials. Raw materials are extremely cheap (and in some cases it took nature a few million years to produce). For example, a barrel of petrol costs about $25 these days, and a copy of Microsoft Office and Microsoft Windows 2000 costs around $700. Which means that for each copy of Office+Windows 2000 the country is paying with 24 barrels of petrol. In general, I believe that we must become software producers (and also technology and innovation producers), and not just consumers. Becoming free software users is a good first step; the next step is to become software producers." (de Icaza)

Another open source pioneer, Ivan Moura Campos, prime developer of Brazil's Popular PC project, believes countries such as Brazil will not overcome the so-called 'digital divide' by relying solely on imported technologies, such as copyrighted proprietary software. As he explains: "[we] realised that this (the lack of access) was not a First World problem. We are not going to find a Swedish or Swiss company to solve this for us. We would have to do it ourselves." (Anderson)

Another example of the problems endemic to closed source proprietary software, especially in operating systems, is that a company such as Microsoft is permitted to bundle a wide number of other computing products into its Windows (and now Windows XP) operating system. Such practices not only capture the global market in operating systems but also control many of the ancillary activities related to day-to-day computing as well. As one commentator has explained:

"The nub of the case against the company is this: why should it be allowed to bundle products like media player into its operating system for "free" instead of being required to distribute them as stand-alone extras?...Who knows how much innovation, especially from smaller firms, has been stifled at birth because of the impossibility of competing with what Microsoft is bundling for free?" (Keegan)

Once again, if software developers in technologically advanced countries such as the US and Europe cannot compete with Microsoft in the application program market, how will software companies in the South? When one proprietary operating system such as Windows becomes the operating standard in countries of the South, no forward internal economic linkages are created and a minimum of wider IT economic development is generated. Any technology that is transferred is primarily an internal transfer to affiliates under its control and the "retention of technology and skills within the network of the trans-national corporations may hold back deeper learning processes and spillovers into the local economy, especially where the local affiliate is not developing R & D capabilities." (UNCTAD, 2001) The principal consequence of this monopolised global proprietary standard in software operating systems is that when companies such as Microsoft expand to new locations in the South, that expansion primarily means the establishment of a local sales office and the hiring of sales and clerical staff and software installers, but few of the much more skilled software writers and programmers. The new Cybercity project in Mauritius is one example of this phenomenon.

5.4 Software Patents and Innovation

Many of the concerns raised in the previous section dovetail with a related question: would a legislative change in countries of the South so as to permit the patenting of software create conditions in these countries which would promote software development and innovation here? This is the main question examined in this section.

As mentioned in Section 3.2, the patenting of software is permitted in some industrialised countries, such as the United States and Japan, and to a somewhat extent because of the 'as such' proviso in Article 52(2) (c) of the European Patent Convention, in Europe. As for the TRIPS Agreement, its legal prescriptions are somewhat less straightforward. Article 27, entitled Patentable Subject Matter, states that with certain exceptions (software is not stated to be an exception), "patents shall be available for *any* inventions, whether processes or products, in *all* fields of technology, provided that they are new, involve an inventive step and are capable of industrial application."(italics added) The wording of this Article takes a broad and non-limiting approach to the patenting of all types of inventions and certainly makes clear that any country that permits the patenting of software is in compliance with TRIPS. But what if a country does not allow the patenting of software? Is it also TRIPS compliant? Certainly one reading, admittedly an expansive reading, of this Article would suggest that software, as an invention and in a field of technology such as software "shall be available" — meaning "should be" or " must be" available — as a patentable invention within the national legislation of all signatory countries. Further, to *not* permit the patenting of software, either explicitly as an excluded category or through the use of "as such" terminology in national legislation, might be interpreted as not in compliance with TRIPS. Yet such an interpretation would have important legal and political consequences. For example, the United States could take a case to the WTO that the United Kingdom was not in compliance with TRIPS because of the "as such" proviso for computer software found in the UK Patents Act, 1977. In the same vein, a wide swath of countries in the South might also face a formal WTO sanction for not explicitly permitting the patenting of software. Yet, at least for the moment, the chances of such a case being made successfully seem slight and a victory on this matter by the United States (or another country) would likely spark a serious backlash.

However, the pressure to require the patenting of software is not abating. Subsequent to the signing of TRIPS, the Unites States has engaged in an ongoing series of bilateral discussions and agreements, often under the rubric of 'free trade', with a range of countries across the South. Requiring such countries to enshrine patents for software in their own domestic laws has been one of the more critical US objectives. For example, under the terms of 2000 memorandum signed between the US and Jordan, Jordan agreed that it "shall take all steps necessary to clarify that the exclusion from patent protection of 'mathematical methods' in Article 4(B) of Jordan's Patent Law does not include such 'methods' as business methods or computer-related inventions." Jordan's patent laws now include this clarification.

Meanwhile, as a result of recent legal changes in the United States — particularly a new set of guidelines created by the US Patents and Trademarks Office for software patents and a 1996 court decision that allowed business method patents — software patents are becoming increasingly common in that country. Each year, the US PTO is granting more than 20,000 patents and such patents now compromise over 15 per cent of all patents issued. (Besson and Hunt) First we need to ask: what is the rationale for this patent policy and does it act as an innovation catalyst for software?

Edith Penrose has provided a number of the commonly stated justifications for patents. The essence of one of the leading arguments for the patent system, that it acts as an encouragement for invention, is this:

"*Industrial progress is desirable. Inventions and their exploitation are necessary to secure industrial progress. Neither invention nor the exploitation of invention will be obtained to any adequate extent unless investors and capitalists have hopes that successful ventures will yield profits which make it worthwhile to make their efforts and risk their money. These profits will not be hoped for unless special measures are taken. The simplest, cheapest and most effective measure is an exclusive patent right in inventions.*" (Penrose)

Given that so much invention is carried out by salaried employees — including in the software field by developers working for large multinational companies — the argument that "capitalists would not introduce innovations or encourage research without the prize of the patent monopoly is the more popular one today." (Penrose)

Penrose concludes that it is "extremely difficult to evaluate this proposition" noting, for example that it is not possible to accurately measure the amount of invention that actually occurs — using the total number of patents filed has been sharply criticised as an accurate measuring device — and there is "no way of knowing what is the 'optimum rate of invention' ". And although a full examination of questions related to the patenting of software — are patents necessary for innovation to occur in software? would the wider introduction of software patents, including in the South, stimulate software development here? — is beyond the remit of this study, some tentative conclusions are possible.

In the first place, the history of computer software reveals that a great deal of development has occurred without any reference to or motivation by the patent system. During the early years (1960's and 1970s) of software development in the US, when important breakthroughs such as UNIX were created through collaborative efforts in American universities, the "sharing by programmers in different organizations of basic operating code of computer programs — the source code — was commonplace…typically no efforts to delineate property rights or restrict reuse of the software were made."(Lerner) In the same vein, the widespread development of free software and its many innovations in the 1980s and 1990s carried out by individual programmers across the globe was not a patent-motivated exercise; collaboration and sharing were —and remain — its two watchwords. Similarly, most recent entries into the open source field, such as RedHat and IBM, have invested large sums of money for R&D, but not applied for patents on their open source software, including on the source code; any attempt to create proprietary code protected by patents would directly contradict the very nature of this alternative system. In short, applying for a patent did not motivate any of the above examples of software innovation.

Whatever the merits of the argument in other technical fields that without patents, innovation would be crippled, various experts in the field of computer innovation have recognized that this is not the way that software has developed. One has noted that "The process of technological advancement in the application programming field, as in many other areas of technology, is through rapid sequential improvements to the existing knowledge base."(Menell) Two noted US economists have conducted a detailed empirical analysis on the relationship between software patents, innovation, and research and development expenditures. They also concluded that software development was 'sequential' and found, by regression analysis, that most US software patents issued are so-called 'cheap patents'.

"We explore whether these patents have increased R&D incentives. We find, instead that software patents substitute for firm R&D; they are associated with substantially lower R&D intensity. Overall, the predominant use of software patents appears to be related to strategic "patent thicket" behaviour."[17] (Besson and Hunt)

A key question that proponent of patenting proprietary software have yet to answer is this: if patenting software actually blocks innovation in the United States, the world's technology leader and a country with most sophisticated IT infrastructure, how would spreading of software patent laws to all countries of the South lead to software innovation in those countries, and particularly by small- and medium-sized industries owned and operated by nationals? Such a global legislative expansion seems destined only to provide further protection within such countries to patented software created within industrialised countries.

A United Nations report has concluded, "many people have started to question the relationship between knowledge, ownership and innovation. Alternative approaches to innovation, based on sharing, open access and communal innovation, are flourishing, disproving the claim that innovation necessarily requires patents." (UNDP 1999) Encouragement of FLOSS in the South is a pressing and contemporary example of how this alternative approach can bear further fruit, as well as contribute to national economic development. If the patenting of software in the United States has lead to "less sharing of knowledge, less innovation and less competition" within that country's software development sector, how will the technology transfer provisions of the TRIPS Agreement lead to more sharing of knowledge with the South? (Smets-Solanes)

5.5 Software Training and Employment Issues

The successful operation of a computer by the average user or the writing of new software by a programmer requires education and training. This section of the report takes up the question of which types of software offers more versatile, portable, and transferable skills in computing. The emphasis here will be on the training opportunities available to those who plan to specialise in computing and the writing of new software programs (and the adaptation of existing programs). It is an issue that is closely connected with the transfer of technology and software innovation. For example, the 1996 UNCTAD draft International Code on the Transfer of Technology suggests that transfer of technology transactions include "[t]he provision of technological knowledge necessary for the installation, operation and functioning of the plant and equipment, and turnkey projects." One commentary on the meaning of this draft Code and technology transfer more generally emphasises that "mere possession of technology does not result in improved technical development or economic gain: the capacity to understand, interact with and learn from that technology is critical." (UNCTAD, 2001)

Across all but the poorest countries of the South, there are an increasing number of training schemes in what is sometimes labelled 'computer sciences'; some training courses (and degrees) are offered at state-operated universities, some courses at stand-alone commercially-operated school or 'computer academies', and some by proprietary software companies. Some closed source proprietary software companies, such as Microsoft, directly operate or fund or make significant philanthropic financial contributions across the globe to a range of software training programmes and schemes, including to those located in the South. Sometimes the contributions can total in the tens of millions of dollars, such as a recent Microsoft contribution to the Mexican government. However, studies have shown that there is an absence of broad computer literacy and technical skills offered by proprietary software training schemes and at many 'computer academies'.(Story) As two Argentinean computer programmers have written:

"The knowledge content of those programs, however, doesn't go any further than providing skills in the use of their proprietary software, and contributes little if anything to the comprehension of the general mechanisms that come into play. They don't teach the user how to use a word processor, for instance, but how to use a very specific, proprietary word processing program. Far from contributing to software literacy, these educational programs are marketing tools designed to produce users that are dependent on a particular program. People who attend these courses are typically unaware even of the existence of alternative solutions, and completely at a loss when confronted with a different program to solve the same need." (Heinz)

Similarly in Africa where 'computer academies' are "a dime a dozen", such institutions "use Microsoft as a matter of course" and graduating students lack broad computing skills. (Buccellato, in Story)

This is a serious problem. Not only is badly needed computer literacy not broadened in less industrialised (and less computerised) countries, but a technological 'bias' is also created. Students end up with a particular type of non-transferable 'skills transfer' from the developed to the less developed world. A well-known saying in the international development movement states: "If you *give* a man a fish, you feed him for one day. If you *teach* a man to fish, you feed him for the rest of his life." The closed source proprietary software training model re-writes that slogan: "Teach a man (or a woman) to fish, but only how to fish in your river and charge annual licensing fees every time he or she wants to put their nets in your water." Such proprietary software training schemes fail to live up the promises made in Article 66 (2) of TRIPS that industrialised countries shall "provide incentives to enterprises and institutions" within least developed countries "for the purpose of promoting and encouraging technology transfer…in order to enable them to create a sound and viable technological base." (TRIPS Art. 66 (2))

The restrictions and limitations encompassed within proprietary software license restrictions (and the overall orientation of proprietary companies to computing issues) work to increase the 'brain drain' from countries of the South to industrialised countries. (See also the comments of Verzola in Section 5.6) As a number of skilled programmers from the South have explained, they often have essentially three career choices:

1. emigrate to industrialised countries where there are many more employment opportunities;
2. stay in their home countries and work as installers of proprietary operating systems and application programs;

3. stay in their home countries and work as programmers on free and open source projects.

As one programmer from Argentina has explained: "as a consequence of widespread use of proprietary software developed abroad, the local market for information technology professionals is limited to openings for 'computer janitor'."(Heinz) Another programmer from South Africa explains that "our human resources are limited in this regard and the last six or so years has seen a huge outflow of computer personnel from South Africa — mostly to Australia or North America. It's simply not an option; access and affordability are just not there."(Buccellato, in Story)

Although further research is needed on this issue, initial evidence suggests that open source-based companies have the potential to create a much higher percentage of skilled jobs (as opposed to merely clerical or sales positions). For example at Linux-based Red Flag Software Co. Ltd. in China, more than 70 per cent of its employees work in some branch of software research and development.(Redflag Software, China) Anecdotal evidence obtained by the author at the January 2004 Idlelo conference in Cape Town, South Africa, which gathered together several hundred programmers and computer experts from across Africa suggests that employment opportunities that take advantage of their advanced technical skills are more available in FLOS-based companies than in proprietary companies operating in Africa.

5.6 Unauthorised Software Copying (Software 'Piracy')

Information (including, but not exclusively, as a form of intellectual property) has the distinguishing characteristic of what economists label a 'public good'.

"Selling information requires disclosing it to others [in most cases[18]]. Once the information has been disclosed outside a small group, however, it is extremely difficult to control…. [As a "public good"] it may be 'consumed' without depletion and it is difficult to identify those who will not pay and prevent them from using the information." (Merges)

The use — and potential misuse — of computer software (and the information contained within it) exemplifies the 'public goods' issue. For a skilled programmer or for even an inexperienced teenager with an Internet connection, copying or downloading a computer program without permission or a licence and distributing it far and wide is a simple matter. Software is a non-rival and quasi non-excludable product and "once access is granted, the software can be copied at almost zero cost."(Osorio) This is true in both Northern industrialised countries and in the South.

There is little dispute that the unauthorised use of computer software, especially proprietary software, is a widespread phenomenon in the South.[19] Indeed, whenever copyright issues in the South are discussed in the mass media and in the business and academic literature of industrialised countries, the unauthorised use of computer software (primarily created in industrialised countries) figures very prominently. But the scope, nature, and consequences of this unauthorised software usage — as well as the solution — are widely misunderstood in most reportage and analysis.

The key elements of the prevailing narrative about the unauthorised copying of software in the South are the following:

- The unauthorised use / illegal copying of proprietary software is widespread. One study conducted by the Business Software Alliance claimed that 92 per cent of the software used in 2002 in China, the world most populous nation, was illegally copied software.(BSA, 2002) Other reports and studies have suggested that 97 per cent of the software used in Vietnam is illegally copied (Carrasco-Muniz, Stocking, BSA, 2003) and one media article, to take a recent example, reported that there is a shop right next door to Vietnam Ministry of Trade in Hanoi which "does a brisk business selling illegal software, movies and music. A pirated copy of Windows and Office goes for no more than (US) $10."(Stocking) In many other countries of the South, rates of unauthorised use and illegal copying of software of at least 80 per cent are often reported; a recent BSA report suggested that 22 of the 25 countries with the highest software piracy rates are located in the South. (BSA, 2003)

By comparison, the rate of illegal copying in Scandinavian countries is about 31 per cent (Osario) and 24 per cent in North America.(BSA, 2003)

- It is assumed that the losses resulting from this unauthorised copying can be accurately quantified … and are extremely large. One 2003 study stated that "[g]lobal dollar losses due to software piracy losses increased 19 % in 2002 to $(US) 13.9 billion" (BSA, 2003) and gives allegedly accurate dollar figures — none are called estimates — for all regions of the world. Even country-by-country and company-by-company figures are reported. For example, a recent news article confidently reported that the widespread copying of software in Vietnam cost Microsoft (US) $40 to $50 million a year.(Stocking) Further the methodology of most such 'piracy' studies assumes that all illegally copied software would have become legal sales if illegal copying were not possible. The Business Software Alliance calculates that "the difference between software applications installed (demand) and software applications legally shipped (supply) equals the estimate of software applications pirated."

- According to the prevailing narrative, the large proprietary software companies are strenuously opposed to this illegal trafficking in software as their short- and long-term interests and profits picture are badly hurt by this phenomenon.

- Finally, it is argued that the main method of combating illegal software copying is strengthened intellectual property laws, nationally and globally, as well as more robust enforcement of existing laws (and widespread educational programs about the evils of illegal copying). Certainly a key motivator behind the inclusion of Article 10 (1) of the TRIPS Agreement and the adoption of the WIPO Copyright Treaty of 1996 was an attempt to reduce unauthorised software copying.

Yet a number of these elements, their reliability, and their meaning are open to challenge from a number of different perspectives.

- Given the fact that most illegal copying is done clandestinely or in the privacy of the home or workplace and can be carried out with relative ease; it is impossible to get an accurate and empirically reliable picture of the extent of this illegal copying. As is well known from criminological studies involving other 'crimes', some crimes are significantly over-reported and others, such as rape and sexual assault, are significantly under-reported. It is simply unknown where illegal copying of software fits along this wide spectrum.

- Even if software 'piracy' investigators obtain accurate data on the number of users using illegal copies in their computers, two Israeli economists demonstrate, in detail, that "this data cannot count for lost sales because there is no indication that all illegal users would purchase [any or equivalent software] from the publisher in the event that copyrights are strictly enforced as to terminate piracy over the Internet." (Gayer and Say)

- Further, what price should be used to calculate actual losses? What if the software in question had been purchased over the Internet and priced at a lower price available in a foreign country? "Even in the case where one can agree on the exact loss of sales resulting from piracy, it is difficult to assess the exact monetary loss since it is not clear which price should be used to value each unit of sale that did not take place." (Gayer and Say)

Hence, even if the quantum of illegally downloaded software is known — and this is widely disputed by a number of economists — this quantum is not equivalent to either lost revenues or lost profits. It is safe to conclude, as do many researchers who have studied the issue in depth, that there is a significant 'over-estimation' of the actual losses suffered, both on a country-by-country and company-by-company basis. Indeed, the calculations become even more skewed when an organisation such as the Business Software Alliance attempts to correlate the global 2002 losses of $US 13.08 billion (due to illegal copying) with the "larger losses in a depressed software market" — how does a depressed market effect the frequency and amount of illegal copying that occurs? — or to conclude that "slightly lower piracy rates" occurred in a year (2002) when there were "generally higher software prices." (BSA, 2003)

The issue is further complicated because:

- Although a proprietary software company may suffer a short-term loss of revenue whenever a software program is illegally copied rather than purchased (or, as is more common, licensed) this company may, in fact, gain significantly in the long run from the short- to mid-tern use of illegally copied software. Why? 'Force-of-habit' and 'what the users are accustomed to' are two of the more important reasons why a computer user (or indeed, the operators of a computer network at a business or an educational institution) will decide upon a particular computer operating system or application program or, just perhaps even more importantly, why they will be reluctant to switch computer systems. 'Switching' systems may cost a great deal of time and money for re-training and re-formatting and installing alternative software; additionally, employee users may be less productive (at least in the short term)… and, in fact, may make employees resistant to change and even downright 'grumpy' about such a forced rupture in their 'normal' computing routines and keyboard movements. Without engaging in conspiratorial thinking, it is fair to conclude that by allowing illegal copying in countries of the South — or often doing little to prevent it — proprietary software companies are allowing and, in fact, encouraging computer users in the South to be 'locked into' one type of computer software system, a proprietary system which, in time, becomes the standard. Such users will certainly be more reluctant at a later moment to switch to an alternative software system, such as FLOSS. And, again at a later moment, when software upgrades are required and enforcement measures are more stringent, proprietary companies can charge higher licensing fees to users. As another economist argues, the evidence suggests "software companies might have a direct and indirect role in helping the generation of illegal copying in underdeveloped countries, and incentives for doing so." (Osorio)

- The use and value of computer software operates, like the telephone, on the principle of 'network externalities'. For certain goods / products based on this principle, "the utility or satisfaction that a consumer derives from the product increases with the number of other consumer of the product.… [In the case of a telephone network], the more people on the network, the more people each person can call and receive calls from."(Merges) Computers operate on a very similar basis; the more people who use the same type of software, the more people who can communicate easily with one another without the need to switch formats and other technical complexities. The second aspect of 'network externalities' is, as mentioned above, that of 'standardisation'. Here an analogy with a standardised typewriter or computer keyboard is helpful. "Because almost all English language typewriters feature the same keyboard configuration, commonly referred to as 'QWERTY', typists need learn only one keyboard system. This standardization enhances worker mobility and the breadth of products available to those who use the QWERTY keyboard."(Merges) In the same fashion, Microsoft Word and Microsoft Word have long been considered and promoted, directly or by default, as the 'standard' systems across many parts of the globe; this standardisation dramatically increases their economic value, especially in the longer term. So in the case of such proprietary software, "network effects are important because, in terms of the total user base, the illegal users add value to all the users, legal and illegal [and those in the South and in the North], and act as the agents in fostering the software's diffusion process by word-of-mouth…[and] indirectly generate additional positive effect for the software company."(Osorio) Various studies and anecdotal evidence reveal that one of the most attractive features of Microsoft products is their global acceptance and prestige as a brand.(Story) Of course, proprietary companies would prefer that users in the South purchase legal software, but the 'short-term pain' of watching illegally copied software being used across the South — and hence becoming the familiar computing and 'lock in' standard and much more valuable because more users world-wide use it — is likely a preferable option for such companies compared to the alternative: for FLOSS to become the global standard. In other words, 'piracy' has its benefits for the diffusion of proprietary software.

- To suggest that mere lawlessness explains why a high percentage of the South computer user illegally copy proprietary software is to overlook a range of other explanations, such as different societal values about the role of private property (including the role of intellectual property, a value system that is often novel in many regions of the South and continues to suffer from severe transplantation problems), peer beliefs that justify copying, and the existing local presence of a particular software system, among a range of factors.

- Finally, there is the question of the relationship between the high costs of proprietary software (see Section 5.1) and national rates/levels of illegal copying. Obviously, there is a relationship and some researchers have shown, for example, that software companies are more likely and more efficiently to reduce illegal copying of their products through price reductions rather than through increased legal enforcement and more severe prosecutions. (Chen and Png). An alternative solution would be to expect all computer users across the South to pay exactly the same price for proprietary software as users do in the industrialised countries and to establish, with much stricter enforcement systems, the same low levels of illegal copying in country such as Vietnam (allegedly 97%) that exist in North America (allegedly 24%). However, the vast differences in per capita income between the two countries suggest this is unrealistic, unless we come to the policy conclusion that most parts of the world should be excluded from computer and Internet era because they are poor and cannot afford proprietary software. Yet, at the same time, it is the very existence of high cost proprietary software and the ease by which illegal copying occurs that feeds the levels of piracy in the South. Two software programmers from, respectively, Argentina and South Africa, give us their views: "The growth-punishing per-seat licenses have encouraged even large companies, and even government itself, to often disregard licensing issues and install irregular software copies in their computers."(Heinz) "Any programme that supports proprietary solutions … feeds the piracy malady which characterises computer usage in South Africa." (Buccellato, in Story)

On the question of illegal copying and its solution in Vietnam, here it is worth quoting the words of Nguyen Trung Quynth, a leading official from Vietnam's Ministry of Science and Technology. Under the terms of a 2001 trade agreement signed with the Unites States, Vietnam is required to reduce its level of illegal software copying. At present, the Vietnamese government is significantly increasing its commitment to Open Source software and, as the centrepiece of this anti-piracy initiative, "[w]e are trying step by step to eliminate Microsoft," said the Vietnamese official. (Stocking)

To contextualize the issue, it also worth remembering that the 'pirating' of the human resources of the South is a historic and continuing phenomenon. This is the response of a Filipino programmer and former copyright commentator to the comment by a representative for U.S. lobbying group Business Software Alliance that "[c]opying licensed software is a form of stealing."

"If it is a sin for the poor to steal from the rich, it must be a much bigger sin for the rich to steal from the poor. Don't rich countries pirate poor countries' best scientists, engineers, doctors, nurses and programmers? When global corporations come to operate in the Philippines, don't they pirate the best people from local firms? If it is bad for poor countries like ours to pirate the intellectual property of rich countries, isn't it a lot worse for rich countries like the US to pirate our intellectuals? In fact, we are benign enough to take only a copy, leaving the original behind; rich countries are so greedy that they take away the originals, leaving nothing behind." (Verzola)

With rare exceptions, piracy is not a problem for FLOSS, especially with Free Software. Under the terms of the GPL licence, the user has the right to improve the code to her/his specifications but all such improvements must be shared with the general pool of users. "Intellectual property is not a part of the business model so piracy is not at issue." (Halbert)

6. FURTHER POLICY IMPLICATIONS AND THE OPTIONS AHEAD

There are a range of legal, economic, technical, and philosophical issues beyond those canvassed in Part Five. This report has concentrated primarily on the background legal regime to the software 'wars' and questions such as the relative costs of the two types of software and technology transfer that are central to economic development in the South. Yet it should not be forgotten that open source software "cannot be viewed as a mere product choice. It reflects more fundamentally an alternative strategy for building, maintaining, and changing the rules that govern information flows in the economy."(Weerawarana and Weeratunga) Indeed some proponents of FLOSS in the South place much less of economic factors and more on matters such as free access to public information by citizens, the permanence of public data, and the security of the state and citizens. For example, to ensure such information access, which is a central objective of Peru's Bill 1609, Free Software in Public Administration, it is "indispensable that the encoding of data is not tied to a single provider", "the goodwill of the supplier" or "on the monopoly conditions imposed by them."(Villanueva) Other studies put more emphasis on how FLOSS promotes social development and collaborative innovation.

This concluding section looks, in summary form, at six issues and conflicts related to proprietary software and FLOSS that are likely to arise in the South in coming months and years and makes a few recommendations, some more firm and definite than others, on a few matters, and suggests some areas requiring further research. It must be underlined that computer software and the question of proprietary vs. non-proprietary systems is still is a new topic on the international North-South development agenda, though certainly some of its sub-themes (e.g. technology transfer) raises issues that remain at the centre of this agenda. At the same time, there are path-breaking FLOSS developments underway across the South and the entire globe may be able to learn some valuable and transferable lessons from the range of projects already established or about to come on stream in the months and years ahead.

For countries of the South, the copyright (and trade secret and sometimes patent) protection accorded to closed source proprietary computer software reminds us, to paraphrase two lines from T.B. MacAulay's well-known 1841 speech to the House of Commons, that: "[t]he principles of computer copyright is this. It is a tax on computer users for the purpose of giving a bounty to multinational proprietary software companies." For hundreds of millions of people living, studying, and working across the South, this 'tax' means that they cannot afford to purchase or use the requisite proprietary software, that they are not given the freedom to modify, and further develop this software for their own particular national or local requirements, and that they will be structurally tied and indebted, both financially and technologically, to industrialised developed countries for decades into the future if proprietary standards become the computing norm across the South. Important reforms are urgently needed. The encouragement and rapid development of non-proprietary systems and installations is a key part of these reforms.

6.1 Proposed Changes to Intellectual Property Laws

As this report has shown, the copyright protection accorded to proprietary software has erected a clear barrier to the widespread use of software across the South. However, any campaign to attempt to change the 'one size fits all' approach of Art. 10 (1) of the TRIPS agreement or to make modifications tailored to the particular computing needs of countries of the South would, at least in the foreseeable future, likely be unsuccessful; such a campaign would be extremely time-consuming and require significant political resources. In any event, bilateral pressures and agreements that likely would be even worse might simply replace the current restrictive multilateral agreements. The computer software industry, especially its dominating US arm, is extremely wealthy and politically sophisticated and has the ability to marshal significant lobbying pressure (e.g. two results are the inclusion of Art. 10 (1) in TRIPS, Art. 4 of the WIPO Copyright Treaty). It would be certain to oppose such changes vigorously and is, at least for the moment, simply too formidable an opponent for countries of the South. Instead of opposing and fighting such copyright restraints, it makes more sense to switch software systems — to FLOSS systems and orientations.

But with patent law, the situation is rather different. Granting patents to software is still limited to a relatively few countries and, unlike the copyright protection afforded software, such patents have no force outside their own national jurisdictions. In the case of software patents, countries of the South should strongly oppose both bi-lateral pressures, especially from the United States, to require the patenting of software in their own domestic legislation any attempts to amend the TRIPS agreement to require the patenting of computer software. Either legal change would be anti-innovative and work primarily for the geographical spread of patent protection of US software from the US to all parts of the globe. This would be of great detriment to countries of the South, including their still embryonic (in most cases) software writing industry of small and medium enterprises and lead to even further global monopolisation of the software industry.

6.2 The Role of Competition/ Anti–trust Law in the Software Field

It is commonplace to argue that competition or anti-trust law have the ability to restrain and correct the tendency of intellectual property regimes to create monopolies. And certainly, given the inherent 'network externalities' and 'standardisation' characteristics of computer software as a technology (see Section 5.6), this tendency to monopoly is especially strong in this field of technology. If a copyright and trade secret protected operating system, such as Windows, establishes a global monopoly — and estimates suggest it currently has 95 per cent of the global PC operating systems market (Keegan) — then this becomes a monopoly that operates outside market pressures or traditional copyright presumptions. Speaking about how its operating system (Windows) was responsible for Microsoft's financial health during the recent downturn in the fortunes of many high-tech companies, a Merrill Lynch high-tech analyst explained that "[s]ince Microsoft has a monopoly in its core business, the company is not vulnerable to the stiff price competition that can hurt other tech leaders in time of weak demand."(Glasner) If this dominant position is problematic in the US, it is even more serious for Southern countries which have a much smaller and even less competitive market and are often dependant on aid packages for computerisation, which sometimes require the use of Microsoft systems. (see Section 6.3)

It should be noted that, in countries of the South, occasional victories have been won by government consumer and fair trading organisations in the software field. For example, as a result of a recent investigation conducted by the Taiwanese Fair Trading office into the costs charged for proprietary software, Microsoft was forced to lower its software licensing prices in that country. But we can safely conclude that such examples are the exception and are likely to remain so for the foreseeable future. The recent anti-trust prosecution of Microsoft in the US is instructive. If the sophisticated anti-trust mechanisms and personnel of the US Department of Justice proved unable to significantly challenge the monopolist practices of Microsoft within the borders of that country, one can hardly expect that the Attorney General's department (or relevant authorities) in countries of the South would be any more successful. In the first instance, actually commencing such an expensive and complex action would not likely be a prosecutorial priority for any of these countries. Moreover, "[c]laims that a rights holder has engaged in anti-competitive behaviour are complex, and resolving them requires significant judicial and legal expertise. Administrative costs may limit a country's ability to undertake competition enforcement…"(World Bank, 2001) Additionally, one can safely predict that such a prosecution would likely be a trigger for a possible Section 301 sanction by the office of the United States Trade Representative. We can conclude then that competition and anti-trust law will generally be impotent in restraining global or national software monopolies. By contrast, FLOSS technology generally raises none of these anti-competitive concerns.

6.3 The Question of Differential Pricing

Using the example of anti-HIV drugs and recent programs to provide such drugs (albeit still in relatively modest amounts in relation to need) to certain countries of the South at prices below those charged in industrialised countries, some studies have recommended that differential pricing should be considered and perhaps implemented for proprietary software (e.g. see the comments in the final report of the UK Commission on IP Rights of Sept. 2002). This is a worrisome proposal. It is one thing to advocate differential pricing (i.e. lower prices for users in poorer countries) for consumer goods, such as pharmaceutical products, that are immediately required to save lives. It is quite another to propose differential pricing for a technology, a functional or utilitarian good, such as proprietary software. Much more than a mere consumer good, software is a key operative technology. Further spread of proprietary software, even if initially at lower prices or 'free', will create further technological 'lock-in' for such countries as has been pointed out at various points in this report.

6.4 The Aid Programs of Industrialised Countries

As one Guinea-based commentator has written, "too often the implementation of foreign aid is all about developing market share and spheres of influence, instead of improving lives."(Marshall) More recently, weaknesses in the information technology infrastructure in the South and the resulting effect of severely limiting Internet access have been identified by industrialised countries, for example at several G8 summits, as 'priority' international development issues. But in the push to computerise the South, it should not be forgotten that, in response to earlier development aid 'priorities', the developing world is today "littered with unused X-ray equipment, broken-down tractors and empty classrooms contributed over the years by well-intentioned and simple minded donors."(Marshall) All computing technology, including software, which is donated to Southern countries by industrialised countries, must be 'appropriate' and adaptable technology for those countries. At the same time, it should be underlined that FLOSS projects can also to be of great benefit in community building and empowerment across the South. As one 2004 study commissioned by the Swedish International Development Cooperation Agency (SIDA) noted:

"*The novel concept of open source software is the notion of community development. With the popularization of the Internet in the late 90s, it became feasible for not just one person or a team in one geographical location, but groups of interested people in geographically dispersed locations to jointly develop software.*" (Weerawarana and Weeratunga)

The aid programs of some nations have often had a poor track record when it comes to the supply of computing technology and software. The US government and, in particular, the Leland Initiative of USAID, which has spent millions of dollars on the expansion of computer access in the South, has taken a rather contradictory view on this matter. On the one hand," we usually purchase PCs and Microsoft products when we furnish systems of this type [for developing countries]," the coordinator of Leland Initiative in Washington, DC, said in 2001 in reply to questions about Leland's software policy.(Story) Yet, on the other hand, this same coordinator explained that, "on balance we are for the cheapest and most affordable approach for the Africans, which would be open source." Development agencies in other industrialised countries, such as the Department for International Development (DFID) in the United Kingdom or SIDA are urged to review whether their own current software aid programmes are aimed at "developing market share" or, conversely, actual technology transfer (see Section 5.3) and technological independence for such countries that FLOSS encourages.

Such agencies could take one immediate step themselves that would signal they are at least open to alternative software systems as DFID, for example, has suggested that it is. One of the more common responses this author heard during this and related research as to why there is still a limited use of FLOSS in Southern nations is this: "if this type of software is so good, why are so many organisations, companies and governments still hooked on proprietary software? Are the alternatives second-rate?" DFID and SIDA could set an excellent example and give an important boost to the status of

FLOSS in the South if they decided to give FLOSS equal-billing in their own external communications and websites … and, in fact, they might decide, as have Amazon.com and the US Pentagon, that using Linux and open source systems could lead to significant cost and efficiency savings over proprietary software.

The costs to government in the industrialised North of supporting the growth of FLOSS in the South are relatively small and the potential benefits to the peoples of such countries are potentially very significant. As one study concluded,"[d]eveloped countries can make cost-effective contributions to less developed countries by helping them adopt free software technologies. Since there is no royalty or per-copy fees, the cost of this transfer is really low for the contributor country. Contributions could be focused in training, localisation, and adaptation to local needs, with a great multiplier effect." (Working Group on Libre Software)

6.5 The Importance of Training Software Technicians

What empowers people are skills. The spread of FLOSS systems in the South will require the establishment of extensive skills programs, both for computer technicians and software writers, as well as for end users of this technology. Mexico's Scholar Net project (see Section 4) provides a negative example for other countries as to what can occur if such a skills base is not provided. Such training programmes are critical for sustainable development.

It is suggested that governments of the South which are keenly interested in joining the information and computing age — and many are — should ensure that sufficient funds are provided for such FLOSS skills and training programs. Not only will such investments work to increase computer access for business and government agencies and provide assistance for the realisation of other goals, such as better educational opportunities, but they will, in time, reduce the flow of foreign hard currency for software to the North. In addition, skills in using FLOSS technology are readily transferable, often at very low cost, and many existing FLOSS project have put the training of new technicians and users at the top of the their priority list.

Leading on from the points made in Sections 6.3 above, development agencies in industrialised countries should consider much more extensive training programs for the development and use of FLOSS technology. Yet, in the end, the active role of national governments in the South remains pivotal. In launching the UN Development Programme's "Human Development Report 2001", its author stated that "[t]he long term solution to innovations for development priorities and conditions of the developing countries will come from the south." (Fukuda-Parr). Closed source proprietary software remains very much a technology of the industrialised North. Facilitating a "south to south" dialogue and the trading of experiences between FLOSS developers, users, and entrepreneurs in Latin America, Africa and Asia would be a worthy international development objective for both governments in the South, as well as European and North American developments, to undertake. The recent Idlelo conference in South Africa provides an excellent example of what can be accomplished by the trading of accomplishment and ongoing problems.

6.6 The Role of the Private Sector and Government Intervention

How the private sector in countries of the South will, more specifically, contribute to the spread of FLOSS is beyond the remit of this report. Other recent studies have proposed various models.(Weerawarana and Weeratunga) Certainly the individual programmers and the private sector, particularly that based in the United States, and companies such as RedHat, SGI and IBM, have played a leading role in the spread of open source technology and these companies have established profitable business models. And while FLOSS technology remains primarily a de-centralised and 'bottom up' approach to innovation — and is likely to remain so — one Brazilian computer expert has written that the idea of an idealised 'loose community' model view of open source development is "highly misplaced. In fact, almost all open source products are produced by a tightly knit group of individuals, usually in at most 2/3 places. Out of more than 400 developers, the top 15 programmers of the Apache web server contribute 88% of added lines."(Camara) While these larger open source companies certainly do not, at present, have the economic and political clout of the multinational proprietary 'giants' and their operational model does not lead to a comparable technological 'lock-in', it is suggested that countries of the South need to closely scrutinise their activities and be vigilant. At the same time, the marketplace in the South is, at least at present, not large enough to support, by itself a whole host of open source companies based in the South and a "government-financed model for open source software is a necessary condition for the production of open source software in the developing world."(Camara)

END NOTES

[1] In this report I have avoided using the commonly-used phrase 'developing countries' because the words "developing countries" are, in my view, misleading. Many remain unconvinced that a number of designated "developing" countries are actually "developing". A recent UN report revealed that 54 countries, mostly from sub-Saharan Africa, were poorer in 2000 than they were in 1900.(Denny) In any event, what does the word "developing" actually mean and is mimicking the process of development followed by "developed countries" a viable or a desirable orientation? Analytic precision is lost as well if, for example, Brazil and Somalia are grouped together in a category called "developing countries". I will instead use the term "countries of the South". This phrase is itself admittedly problematic and I add the vigorous caveat that there is a great disparity among, between, and within such countries; the place of South Africa and the different South Africas within Africa is a good case in point. "The concept 'development', like the concept 'growth' — both are borrowed from biology — is a natural metaphor meant to obscure and obfuscate the violence and crude exploitation that continue to characterize the relationship between 'development' and 'underdevelopment'."(Mies)

[2] Throughout this report, the terms Free Software and Open Source Software (OSS) are used interchangeably, except where a distinctions is required to highlight particular differences. Some elements of FLOSS may be proprietary, that is, owned as a private property right that is protected by global and domestic intellectual property laws, particularly copyright laws. Readers not familiar with basic computing concepts are referred to Appendix 1: Definitions of Seven Key Computing Terms.

[3] Microsoft has labelled open source software an "intellectual property destroyer" that is un-American and based on "unhealthy" principles.(Charny) Other companies, such as the US-based SCO are becoming increasingly shrill in their denunciation of open source software and linking the issue to the war on terror. Its chief executive reportedly wrote that "Open Source software — available widely through the Internet — has the potential to provide our nation's enemies or potential enemies with computing capabilities that are restricted by US law. A computer expert in North Korea who has a number of personal computers and an Internet connection can download the latest version of Linux, complete with multi-processing capabilities misappropriated from UNIX, and, in short order, build a virtual supercomputer … The unchecked spread of Open Source software, under the GPL [General Public Licence] is a much more serious threat to our capitalist system than US corporations realize."(Orlowski)

[4] According to a recent study of world population trends, "the growth of the human population has been, is now, and in the future will be determined in the world's less developed countries … Ninety-nine percent of global natural increase — the difference between numbers of births and number of deaths —now occurs in the developing regions of Africa, Asia and Latin America." (U.S. Department of Commerce, 1999)

[5] Obviously the above figures are slightly out of date, but the overall pattern remains.

[6] If you are not familiar with the meaning of basic computer terms such as "operating system" and "source code" open source", you may wish to read Appendix 1:Definitions of Seven Key Computer Terms before proceeding.

[7] Copyright protects a bundle of exclusive legal rights granted by national legislatures to the owners of expressions (works), which include computer software. These rights include the exclusive right to make copies of the work and to distribute the work. Protection is essentially automatic upon creation and fixation of the work; there are no formal registration requirements.

[8] A patent is a monopoly right granted by national legislatures to protect inventions, either the product or the process of making that product. The grant of a patent requires compliance with a formal registration process; to be patentable, an invention must be novel, involve an inventive step, be capable of industrial application and not fall within one of the excluded categories.

[9] 'Trade secret' means information, including a formula, pattern, compilation, program, device, method, technique or process, that: (1) derives independent economic value, actual or potential, from not being generally known to,

and not being readily ascertainable by proper means by, other persons who can obtain economic value from its disclosure or use, and (2) is the subject of efforts that are reasonable under the circumstances to maintain its secrecy. (US Uniform Trade Secret Act.)

[10] Intellectual property is an intangible form of property and has a number of important differences with various forms of physical property, such as the fact that it is non-rivalrous in consumption. Collapsing these distinctions leads to *extremely misleading* analytic and policy conclusions.

[11] Cohen based this 'future distributions' insight on the ownership of land and machinery, but it works equally well in the case of intellectual property. In fact, given the economics of the licensing of IP (that is, most licensing income goes straight to a company's 'bottom line' without further investment or expenditure) ownership of copyright is even more determinative of future acquisitions than ownership of machinery. For example — and here taking an example from outside the field of software — the lyrics to the song 'Happy Birthday to You' (written in 1893, copyrighted in 1935, expiration date of 2021) have already earned copyright royalties of £UK 39m and the lyrics still have nearly another 20 years to go as an earner for Warner Communications, the current owner. See A. Salamon, 'On the Other Hand, You Can Blow Out the Candles for Free', WALL STREET JOURNAL, 12 June 1981.In the same vein, the future potential earnings of copyright-protected Microsoft or Sun products is essentially incalculable.

[12] For a further explanation and critique, see Alan Story, "Burn Berne: Why the Leading International Copyright Convention Must be Revealed" 40 UNIVERSITY OF HOUSTON LAW REVIEW 763 (2003).

[13] The material in this section has been compiled from a wide range of current, primarily Internet based, media sources as well as from e-mails and interview notes received and prepared by the author over the last 24 months. Because of this complexity of sources, I have not given citations for all of the developments noted here, but can provide, on request, sources for any particular items. There have also been many important FLOSS developments in rich industrialised countries, but such countries are outside the parameters of this study and have been omitted.

[14] The final draft of this report was completed in January 2004 and I have attempted to present information and facts that were current and accurate as of that date.

[15] This study also correlates 'piracy' rates for these countries, but this calculation is omitted from the figures presented here.

[16] Thanks to Prof. Victor van Reijswould and the computer staff at Uganda Martyrs University for assisting me in making these calculations.

[17] 'Patent thicket' behaviour refers to attempts by a company to build a thicket of patents which it may itself not use and which are primarily aimed at limit the ability of competitors to enter the same technological market. Such patents are sometimes also called 'blocking patents'.

[18] Significantly, proprietary software companies do not disclose the source code, its own vital information when such software it is licensed to a user.

[19] This phenomenon is commonly called 'software piracy' but this misleading and pejorative term is avoided here; 'piracy' means "the crime of a pirate" and properly refers to the sinking of ships at sea, often with loss of life and property. As one commentator has noted, "the 'piracy' concept is a rhetorical device extending property discourse into previously unclaimed territory." (Kretschmer)

APPENDIX 1: DEFINITIONS OF SEVEN KEY COMPUTER TERMS

The following definitions of seven key computer terms are taken, in an edited form, from *What is Techtarget*, available at: http://whatis.techtarget.com/whome/0,289825,sid9,00.html

Although both the design and operation of modern computer platforms are obviously complex technical matters, a basic grasp of these seven terms should be sufficient for the 'non-geek' layperson to understand the main points that this report is attempting to make.

1. Hardware (and software)

Hardware is the physical aspect of computers, telecommunications, and other information technology devices. The term arose as a way to distinguish the 'box' and the electronic circuitry and components of a computer from the program you put in it to make it do things. The program came to be known as the software. Hardware implies permanence and invariability while software or programming can easily be varied. You can put an entirely new program in the hardware and make it create an entirely new experience for the user. You can, however, change the modular configurations that most computers come with by adding new adapters or card that extend the computer's capabilities. Like software, hardware is a collective term. Hardware includes not only the computer proper but also the cables, connectors, power supply units, and peripheral devices such as the keyboard, mouse, audio speakers, and printers.

2. Operating system

An operating system (OS) is the program that, after being initially loaded into the computer by a boot program, manages all the other programs in a computer. The other programs are called applications or application programs. The application programs make use of the operating system by making requests for services through a defined application program interface. An operating system performs these services, among others, for applications:

- it determines which applications should run in what order and how much time should be allowed for each application before giving another application a turn;
- it manages the sharing of internal memory among multiple applications;
- it handles input and output to and from attached hardware devices, such as hard disks, printers, and dial-up ports.

All major computer platforms (hardware and software) require and sometimes include an operating system. Linux, Windows 2000, VMS, OS/400, AIX, and z/OS are all examples of operating systems.

3. Application Program

An application program is any program designed to perform a specific function directly for the user or, in some cases, for another application program. Examples of application programs include word processors; database programs; Web browsers; development tools; drawing, paint, and image editing programs; and communication programs. Application programs use the services of the computer's operating system and other supporting programs.

4. Source (and object) code

Source code and object code refer to the 'before' and 'after' versions of a computer program that is compiled before it is ready to run in a computer. The source code consists of the programming statements that are created by a programmer and then saved in a file; this file is said to contain the source code. The resulting output (or compiled file) from the source code file is often referred to as object code. The object code file contains a sequence of instructions that the processor can understand but that is difficult for a human to read or modify. For this reason and because even debugged programs often need some later enhancement, the source code is the most permanent form of the program. When you purchase or receive operating system or application software, it is usually in the form of compiled object code and the source code is not included. Proprietary software vendors usually don't want you to try to improve their code since this may created additional service costs for them. Lately, there is a movement to develop software (Linux is an example) that is open to further improvement and here the source code is provided.

5. Proprietary (as in 'Proprietary software')

In information technology, proprietary describes a technology or product that is owned exclusively by a single company that carefully guards knowledge about the technology or the product's inner workings. Some proprietary products can only function properly if at all when used with other products owned by the same company. (An example of a proprietary product is Adobe Acrobat, whose Portable Document Format (PDF) files can only be read with Acrobat Reader.) Microsoft is often held up as the best example of a company that takes the proprietary approach. It should be observed that the proprietary approach is a traditional approach. Throughout history, the knowledge of how an enterprise makes its products has usually been guarded as a valuable secret and such legal devices as the patent, trademark, and copyright were invented to protect a company's intellectual property.

A prime motivation behind development of products using proprietary technology is straightforward: buyers are compelled to use other products marketed by the same company. Nevertheless, the strongest reason in favor of using proprietary standards leads to the strongest reason against: customers may be disinclined to buy a product that limits their choice of others. (Proprietary software is also sometimes called 'closed source software' (CSS))

6. Open source software

Open source software (OSS) refers to software that is developed, tested, or improved through public collaboration and distributed with the idea that the source code must be shared with others, ensuring an open future collaboration. The collaborative experience of many developers, especially those in the academic environment, in developing various versions of the Unix operating system, Richard Stallman's idea of Free Software Foundation, and the desire of users to freely choose among a number of products - all of these led to the open source movement and the approach to developing and distributing programs as open source software.

7. Free software

Free software is software that can be freely used, modified, and redistributed with only one restriction: any redistributed version of the software must be distributed with the original terms of free use, modification, and distribution (known as copyleft). Free software may be packaged and distributed for a fee; the 'free' refers to the ability to reuse it, modified or unmodified, as part of another software package. As part of the ability to modify, users of free software may also have access to and study the source code. The best-known example of free software is Linux, an operating system that is proposed as an alternative to Windows or other proprietary operating systems. Free software is easily confused with freeware, a term describing software that can be freely downloaded and used but which may contain restrictions for modification and reuse.

REFERENCES

International agreements and conventions and national statutes

The Berne Convention for the Protection of Literary and Artistic Works (Paris, 1971).

Agreement on Trade-Related Aspects of Intellectual Property Rights (TRIPS Agreement) 1994.

The UK Copyright, Designs and Patents Act, 1988 (as amended).

The US Copyright Act, 1976 (as amended).

The US Digital Millennium Copyright Act, 1998.

WIPO Copyright Treaty, 1996.

US Uniform Trade Secret Act.

Books

Drahos, P.F. "A Philosophy of Intellectual Property", Aldershot, Dartmouth, (1996).

May, C. "A Global Political Economy of Intellectual Property Rights", Routledge, (2000).

Merges, R. et al "Intellectual Property in the New Technological Age", New York, Aspen Law and Business, (1997).

Moody, G. "Rebel Code: Linux and the open source revolution", London, Allen Lane, The Penguin Press, (2001).

Penrose, E. "The Economics of the International Patent System", (1951).

Ricketson, S. "The Berne Convention for the Protection of Literary and Artistic Works: 1886-1986", London, Centre for Commercial Law Studies, (1987).

Articles, Reports, and Other Materials

Anderson, R. "Low-Cost Computers for the People", Benton Foundation, (27 August 2001), http://www.digitaldividenetwork.org/content/stories/index.cfm?key=178.

Ard, S. "Microsoft Memo: Linux fight backfiring", CNET NEWS, (6 November 2002).

Barlow, J. "Selling Wine Without Bottles: The Economy of Mind on the Global Net", WIRED MAGAZINE, (1994).

Benkler, Y. "Coase's Penguin, or Linix and the Nature of the Firm", Unpublished paper presented at "The Public Domain" conference, Duke Law School, Durham, N.C. USA, (November 2001).

Bridges.org, "Spanning the Digital Divide - Understanding and Tackling the Issues", (May 2001), http://www.bridges.org/spanning/.

Buckman, R. "A Titan's Power", WALL STREET JOURNAL, (29 June 2001).

Business Software Alliance, "Global Software Piracy Study- Asia/Pacific, 2002", http://global.bsa.org/globalstudy/asia/.

Business Software Alliance, "Eight Annual BSA Global Software Piracy Study", (June 2003), http://global.bsa.org/globalstudy/2003_GSPS.pdf.

Camara, G., National Space Research Organisation, Brazil, (e-mail on file with author).

Carrasco-Munoz, J. "Free/Libre/ Open Source Software as official development aid", Paper delivered at eGOVOS Conference, Washington, DC, USA, (16-18 October 2002), (on file with author).

Charny, B. "Microsoft raps open-source approach", CNET News, (3 May 2001).

Chen, Y. and Png, I "Software Pricing and Copyright Enforcement", National Taiwan University, (2001).

Cohen, M. "Property and Sovereignty", CORNELL LAW REVIEW, XIII, 8 (1927).

Computer Aid International website, http://www.computer-aid.org/.

Computer Professional for Social Responsibility, "Shaping the Network Society", DIAC-02 Symposium, Seattle, Washington,USA, (16-19 May 2002), http://www.cpsr.org/conferences/diac02/.

Cooper, M.N. & Murray, C. "Windows XP/.Net: Microsoft's Expanding Monopoly – How It Can Harm Consumers and What the Courts Must do to Preserve Competition", Consumer Federation of America, Consumers Union, Media Access Project, U.S. Public Interest Research Group, (26 September 2001).

Correa, C. "Review of the TRIPS Agreement: Fostering the Transfer of Technology to Developing Countries", Third World Network (2001).

Cross, M. "Inside IT: High stakes in the battle for Britain", THE GUARDIAN, (29 January 2004).

Cundiff, V.A. "Protecting Computer Software as a Trade Secret", 507 Practicing Law Institute, 'Patents, Copyrights, Trademarks and Literary Property Course Handbook Series', 761, (1998).

Denny, C. "UN Fears Iraq will dominate summit", THE GUARDIAN, (30 May 2003).

Deutsch Bank Research, "Economics – Internet revolution and new economy", Frankfurt am Main, Germany, (November 2002).

The Economist, "Extending its tentacles", (18 October 2001).

Festa, P. "Governments push open-source software", CNET NEWS, (29 August 2001), http://news.cnet.com/news/0-1003-200-6996393.html?tag=tp_pr.

Festa, P. "South Africa considers open source", CNET NEWS.COM, (5 February 2003).

Fokkena, L. "Like Breast Milk and Goat Poop: The Case for Using Open Source Software in the Third World", Kite Inc., (2001), http://www.kiteinc.org/Goats.

Fukuda-Parr, S. "Making new technologies work for human development", in HUMAN DEVELOPMENT REPORT 2001, United Nations Development Programme, London, (10 July 2001), http://www.undp.org/hdr2001/sakiko.html.

Gayer, A. and Shy, O. "Internet, Peer- to- Peer, and Intellectual Property in Markets for Digital Products", University of Haifa, Israel, (4 May 2002), http://www.wiwi.hu-berlin.de/wt1/lectures/mikroseminar/0203/freeware19.pdf .

Gervais, D.J. "The TRIPS Agreement- Interpretation and Implementation", EUROPEAN INTELLECTUAL PROPERTY REVIEW 156, (1999).

Ghosh, R.A. "Licence fees and GDP per capita: the case for open source in developing countries", FIRST MONDAY, Vol.12, No. 8 (December 2003).

Glasner, J. "A Good Time to Be a Monopoly", WIRED NEWS, (18 October 2001).

Halbert, D. "Asian Futures: Piracy, Open Source and the International Intellectual Property Law", (September 2001), (Unpublished; on file with author).

Heinz, F. et al. "Proprietary Software and Less-Developed Countries: The Argentine Case", Appendix 3 in A.Story, CIPR, (2002).

de Icaza, M. "Miguel de Icaza Tells All!", Slashdot, (4 April 2001), http://slashdot.org/interviews/00/04/03/2344211.shtml.

The International Intellectual Property Alliance, "Lebanon Report 301/99", (1999), http://www.iipa.com/rbc/1999/rbc_lebanon_301_99.html.

International Telecommunications Union, "Internet Indicators" (2002).

Keegan, V. "Hand a rum deal to Gates", THE GUARDIAN ONLINE, (2 November 2001).

Kretschmer, M. "Piracy Revisited", Issue paper for the Social Sciences Research Council conference on Intellectual Property, Markets and Cultural Flows, (24-25 October 2003).

Lerner, J. and Tirole, J. "The Simple Economics of Open Source", National Bureau of Economics Research Working Paper 7600, (March 2000).

Levinson, M. "Let's Stop Wasting $78 Billion a Year", CIO MAGAZINE, (5 October 2001), http://www.cio.com/archive/101501/wasting.html.

Marshall, W. "Alogrithms in Africa", LINUX JOURNAL, (18 May 2001).

Menell, P.S. "An Analysis of the Scope of Copyright Protection for Application Programs", 41 STANFORD LAW REVIEW 1045, (1989).

Menell, P.S. "The Challenges of Reforming Intellectual Property Protection for Computer Software" 94 COLUMBIA LAW REVIEW 2644, (1994).

Mies, Maria, "Gender and Global Capitalism', in L. Sklair, ed. CAPITALISM AND DEVELOPMENT, 107, (1994).

Newsweek, "Free for All; open-source software transforms technology in the developing world", (30 June 2003).

Oddi, S. "The International Patent System and Third World Development; Reality or Myth", DUKE LAW JOURNAL 831, (1987).

Okediji, R.L. "Development in the Information Age: Issues in the Regulation of Intellectual Property Rights, Computer Software and Electronic Commerce", Issue paper No. 9 of the UNCTAD-ICTSD project on IPRs and sustainable development, (May 2004), http://www.iprsonline.org/unctadictsd/docs/.

Orlowski, A. "Open Source thieves stealing my American code – SCO boss", THE REGISTER, (22 January 2004).

Osorio, C.A. "Illegal Copying and Diffusion of Software in Developing Countries: Empirical and analytical evidence on the conquer of emerging markets' software markets", (2003), (Unpublished paper on file with the author).

Otter, A. "NGO launches long-term open source research", Tectonic- the source for open source, (17 January 2003).

Panos.org.uk, "The Internet and the South: Superhighway or Dirt-Track?", Panos Media Briefing No. 16, (October 1995), http://www.oneworld.org/panos/briefing/internet.htm.

Peeling, N. and Satchell, J. "Analysis of the Impact of Open Source Software", (October 2001) http://www.govtalk.gov.uk/rfc/rfc_document.asp?docnum=429.

Primo Braga, C.A. and Fink, C. "The Economic Justification for the Grant of Intellectual Property Rights: Patterns of Convergence and Conflict", 72 CHICAGO-KENT LAW REVIEW, 439, (1996).

Rebelo, P. "Brazil Counting on a Net Gain", WIRED, (23 February 2001), http://www.wired.com/news/culture/0,1284,41785,00.html.

Redflag Software Co. Ltd., "About us", (undated).

Reuters, "Signs of progress seen in China's anti-piracy push", (16 October 2001).

Reuters, "Mac, Linux, Even DOS fans Diss XP", WIRED NEWS, (26 October 2001).

Roffe, P. and Tesfachew, T. "Revisiting the Technology Transfer Debate: Lessons of the new WTO Working Group", BRIDGES Vol. 6, No. 2, (2002), http://www.ictsd.org/issarea/ag/products/Development%20Box.pdf.

Scheeres, J. "Peru Discovers Machu Penguin", WIRED NEWS, (22 April 2002).

Shankland, S. "Korean Government buys into Linux", CNET NEWS, (16 January 2002).

Siwek, S. "Copyright Industries in the U.S. Economy – The 2002 Report", Economists Incorporated and the International Intellectual Property Alliance.

Smets-Solanes, J-P "Stimulating competition and innovation in the information society", (March 2001), http://www.pro-innovation.org/rapport_brevet/brevets_plan-en.pdf.

Stikeman, A. "PicoPeta Simputers", TECHNOLOGY REVIEW, (September 2001).

Stocking, B. "Vietnam embracing open-source products, Nations solution to software piracy: eliminate Microsoft", MERCURY NEWS, (30 October 2003).

Story, A. "Copyright, Software and the Internet Copyright", Study Paper 5, Commission on Intellectual Property Rights, London, (2002), http://www.iprcommission.org/graphic/documents/study_papers.htm.

United Nations Conference on Trade and Development, "Transfer of Technology", United Nations, (2001).

United Nations Development Program, "Human Development Report 1999", Oxford University Press, (1999).

United Kingdom Cabinet Office, "Open Source Software- Use within UK Government", Draft for Public Consultation, (10 December 2001).

U.S. Department of Commerce, Economics and Statistical Administration, Bureau of the Census, "World Population at a Glance: 1998 and Beyond", IB/98-4, (January 1999), http://www.census.gov/ipc/prod/wp98/ib98-4.pdf.

van Reijswould, V. "Comparing Open Source Software: An Overview of Some Aspects", (January 2003), (on file with the author).

Verzola, R. "Pegging the World's Biggest", 12 EARTH ISLAND JOURNAL 41, (1997), http://www.earthisland.org/eijournal/new_articles.cfm?articleID=311&journal .

Villanueva, E. "Open letter to Microsoft, Peru", (8 April 2002).

White, E. "Making the Software Industry 'Open'", Cooperation South #1, (2001).

Wildstrom, S.H. "Windows XP: Time to Buy a New PC?", BUSINESS WEEK, (8 October 2001).

World Bank, "Global Economic Prospects and Developing Countries 2002", Chapter 5: Intellectual Property: Balancing Incentives with Competitive Access, (2001).

World Summit on the Information Society, "Draft Action Plan", WSIS/PCIP/DT/2-E, (21 March 2003).

Working group on Libre Software, "Free Software / Open Source: Information Society Opportunities for Europe?", (Version 1.2 - work in progress), (April 2000), http://eu.conecta.it/paper/paper.html.

UNCTAD-ICTSD project on IPRs & Sustainable Development: Publications