Alternative Routes in the Digital World

Open Source Software in Africa [1]

Victor van Reijswoud
Department of Information Systems
Uganda Martyrs University
Nkozi – Uganda
victor@umu.ac.ug
+256 77 908490

Corrado Topi
School of Computing and Engineering
University Of Huddersfield
Huddersfield - West Yorkshire – UK
c.topi@hud.ac.uk
+44 1484 47 3700

"The box said that I needed to have Windows 98 or better...
so I installed Linux."
--- CARUS M. (221556)

There, I've said it. I'm out of the closet. So bring it on...
--- Linus Torvalds

Quotes on: http://www.ao.com/~regan/quotes/Linux.html

Abstract

Software allows people to work with computers. Operating Software controls the hardware components and application software provide tools to facilitate and support the users' work. Most of the softwares are owned by private people or companies and users by licenses to use the software. This type of software is called proprietary or closed source software since the user purchases a license for using the product and the actual product (source code). At present Microsoft and Oracle are the biggest producers of this type software in the world. In the two decades a new approach for software development is emerging. Open Source Software movement is built on the premise that better software is produced when everyone is allowed to modify and change the software. So, in stead of selling user licenses, the product (source code) is distributed.

[1] This research was greatly sponsored by the Dutch IT and Management Consulting firm VIA Groep nv (www.viagroep.nl) to support research in the East African region.

The article discusses the differences between Open and Closed Source Software and reasons that organizations in the African context should decide to embrace the Open Source Software initiative. Several emerging initiatives promoting the use of Open Source Software are considered.

1. Introduction

In 1991 Linus Torvalds introduced a new paradigm in software development that is now maturing and has the potential to change the world. Torvalds developed an operating systems called Linux. Initially he was interested in developing a small version of the UNIX operating systems. In order improve the software he decided to share the code with the software community outside the University of Helsinki in Finland. The software community based approach in the development of Linux gave the real boost to the Open Source Software (OSS) philosophy, since it was proved that it was able to produce software that was able to compete with commercially produced softwares (www.linux.org). The launch of the first Linux distribution (a combination of the operating systems and supporting applications) by Torvalds in 1994 has lead to an explosion new Linux based Open Source operating systems and application software to run on the Linux platform. At the moment of writing www.linux.org lists 185 different Linux distributions[2].

The Open Source Software philosophy challenges the general accepted software development paradigms that are used by companies of today (http://cruel.org/freeware/cathedral.pdf). Traditional software development paradigms are based on the idea that software has to be fully developed and tested before it is sold in the market. When the software is put in the market, users can not change the source code, and mistakes have to fixed by the software company. This way of working makes the development of new software a labor intensive and long process. With the development of Open Source Software, a different route is taken. The basic functionality is programmed by the initiator(s) and then made available for others to test, use and/or modify. Mistakes in the software are not considered problematic, but are accepted. Since the source code is distributed, every software engineer can change or extend the original product. So, where propriety software is developed in-house and then released, Open Source software is under constant development because anyone in the world can change the code (http://www.gnu.org and http://www.fsf.org).

An important aspect in pro-OSS discussions is the price. Not all OSS is free, but in most cases it is cheaper to acquire than proprietary software. The real price difference emerges from the fact that

[2] Accessed July 2nd 2003.

there not a license fee structure. Where for proprietary software all the users need to pay a fee, in the OSS approach someone buys the software, and becomes the owner and can start to freely redistribute it to other users. Especially in larger organizations this can make a huge difference. Although a lot of emphasis is put on this aspect of OSS, but as we will there are more aspects that are relevant in the African context.

The purpose of this article is to elaborate on the role of Open Source Software can play and plays in the African context. Since there is little tradition in computing, OSS can find a good breeding ground in Africa. On the other hand, software engineering in Africa can greatly benefit from the expertise that is made available by the OSS community. In the paper we start with elaborating on some of the relevant aspects of the Open Source Software initiative and decribe some of the advantages and disadvantages of of the adoption of Open Source Software. We then shift the focus to the African continent. We investigate some African OSS initiatives. Finally we try to answer the question why OSS is important for Africa? With this article we hope to stir up the awareness of policy makers in Africa, because some changes in software policy are needed.

2. Open vs Closed Source Software

The discussion about Open and Closed Source Software raises a growing variety of issues the moment more disciplines join the debate. In the current section we focus on three aspects that are directly relevant for the African context: licensing policies, the software development process, the software market situation and we end the section with a discussion of the advantages and disadvantages of Open and Closed Source Software.

2.1 Licensing policies

Software is an essential element in the operation of every computer, from PDA to notebook, from desktop to server. At a general level we identify two types of computer software: operating systems software and application software. We could introduce more complex classification of the different software layers, as the OSI model, but they will be beyond the scope of the present paper. Beyond our scope is also to define the boundaries, which separate the operative system software from application software. We would then accept the following broad definition of operative system software

Operating systems software is designed the make all the different hardware components in the computer, as well as all the peripherals, work together and operate as an integrated machine. The

operative system does interpret signals from keyboard and other input peripherals, allowing the user to input data, to process it in the central processing unit, store it temporarily or permanently on storage devices, and provide an output on output peripherals, as screen or printer. At present modern operating systems are e.g. Microsoft Windows XP and Red Hat Linux. Application software, on the other hand, is designed to support individual users or organizations in executing their tasks. Application software is used on top of the operation systems software. There are many types of application softwares and even more manufacturers. For office applications, like word processing and spreadsheet software, users are mostly dealing with Microsoft Office, Corel Office suite, StarOffice or Open Office. For database applications we see that the leading applications are provided by Microsoft (SQL Server), Oracle and MySQL. The list is long, there are applications for almost every thinkable task end multiple companies that offer software solutions for each of these tasks. Proprietary Software is essentially software for which the user needs to pay royalties for using and does the user not allow to make modifications to the software code. In most cases the company that produces the software takes all possible measures to prevent users or software engineers to reveal and modify the program code.

Another distinction that is useful for our discussion is the distinction between software that is used at the server side and software at the users' end. When using computers in a networked environment, the user is only confronted with a small proportion of all the software that is used. To connect and survive in a computer network the user is connected to one or more servers that house information and software. Without going into detail we can say that server-side software regulates the interconnection of the computers in so-called a Local Area Network (LAN) that connects the computers in an organization, or in a Wide Area Network (WAN) when different locations of an organization are connected, or on the Internet when the public domain network is used. For the user this software is mostly invisible and applications on the user side are used to navigate through the network without knowing the networking details. The most commonly known way of navigating through the network are Windows Explorers or Internet Browsers. On the server-side, which is mostly operated by the network administrator or network operator a lot of different applications and hardware are used. We can identify for example software to connect a server to the Internet, or to send an retrieve our electronic mail or to allow files to be transferred from one computer to another. However, in most cases the user is not even using the possibilities of a network. S/he is just using applications that support the office tasks at hand.[3]

[3] For the version of the African Research and Development issue I prefer to keep this explanation in the text. Not everyone is knowledgeable about these aspects.

When we consider the licenses of the software that is available in the market, we identify two licensing paradigms: Open Source Software (OSS) and Closed Source Software (CSS).Although both use licenses to protect the ownership of the software, they greatly differ in the extent to which they protect the rights to modify and redistribute and sell the product as well as the underlying software code.

In the OSS paradigm the source code is made openly available on the market. Up to a certain extent and depending on the type of OSS license, the user becomes the *owner* of the source code and of the binaries, and can modify it, reuse it in order to build other packages, fix bugs, make the source code and the binaries available to other users. The user, who owns the package, can legally install it on an unlimited number of computers, and eventually, if possible, even recompile it for different platforms. Depending on the type of license in use, some minor constraints can be applied.

In the CSS paradigm the code of the software is not made available to the outside world. Only the binaries, also known as executable of the package are available on the market, and the user does only own the legal right *to use* the package, but not the source code and not even the binaries. Even the operation of fixing bugs could be considered illegal under this paradigm. Attempts to rebuild the source code, using re-engineering methods and tools are illegal. Generally speaking, the user can legally install the package only on a limited number of computers (basically, one), and cannot recompile it or migrate it to different platforms. Further installation would be considered illegal.

Both the CSS and the OSS licenses know a lot of different versions.

Within the CSS community, it is normal practice that each software producer designs their own license that goes with the software. Large software companies like Microsoft and Oracle has specially designed user licenses, smaller organizations mostly work with standardized licenses to protect the intellectual property of their software. For more details on the licensing of Microsoft see http://www.microsoft.com/licensing/ and for Oracle software more information can be found at http://www.oracle.com/corporate/pricing/.

Within the OSS community, we identify two major trends in licensing: Open Source licenses and Free Software (FS) licenses.

FS licenses are the OS approach in its stronger form. FS licenses propagate indeed complete freedom to use the software's source code for any purpose and in any environment. The user of the packages released under FS licenses are granted complete access to the source code, as well as the right to all modification, to redistribute copies so that you can help your neighbour, and to improve the software and to release the improvements to the public so that the community can benefit. No constraints are allowed, and FS licenses in its strongest form, the GNU GPL license, are self-

propagating, *id est* every modification to the source code of a package, which had originally been released under GPL, must be released under the same license. A package released under GPL can only evolve and be used in other packages, which are released under GPL themselves. The details for Free Software licenses are defined in the GNU Manifesto (for details see www.fsf.org) and under this license high quality software has been produced since 1984.

The OS licenses are defined by the Open Source Definition. The Open Source Definition builds on the GNU Manifesto, but tries to provide credits to the originator of the software and to protect a product that is already in the market from misuse. At present there are more than 30 different licenses that are harboured by the Open Source Initiative. They differ from each other in the extend to which modification, redistribution and (re)selling of the software is allowed. Most important, packages released under some of the OS licenses, which still comply with the Open Source Definition, do not necessarily have to be released under the same license. Theoretically, a package released under CS license could then be built on top of another package released under OS license, even if the original OS licensed package has to be distributed along with the added CS components[4].

In order to get an understanding of the nature of Open Source licenses, the Open Source Definition as maintained by the Open Source Initiative is added in appendix 1 of this article (for further details on different OSS licenses see: www.opensource.org).

Notwithstanding these secondary differences, FS and OS licenses are perfectly compatible, and FS licenses are indeed all Open Source Definition compliant, *id est* FS licenses are all also OS licenses, whilst the opposite is not true.

2.2 Software development processes

The difference between Open Source and Closed Source Software paradigms is deeper than just the fact that the software is 'free' or 'open'. One of the most important result of the underlying assumptions of these two approaches, is the difference in the way software is developed, that is in the software development process (SDP).

Erik Raymond, in his book *The Cathedral and the Bazaar* (1998), tried to illustrate the two paradigms by using the metaphors of the 'Cathedral' and the 'Bazaar'.

The cathedral metaphor symbolizes commercial software development. The software development process is based on a thorough analysis and design specification and developed in a closed environment with dedicated software engineers. The creativity of the software engineers is

[4] Examples: Ximain or Mac OS X

strongly controlled by the pre-determined design. Creativity is put into the design by the software designers, while engineers are used to code plans. Testing plays an important part in the cathedral paradigm. The produced software package is then thoroughly tested, in an attempt to eliminate the number of known critical bugs before the software is released to the public. Bugs are considered a shortcoming of the software and if too many bugs are found in proprietary software, software is considered to be of low quality. Once released, a new or updated version will be prepared in the same way. When a significant amount of bugs are discovered, or bugs with serious consequences, like security risks, the software producer releases a patch to solve the problems. In some cases the software company has to issue several patches before a new, and improved version of the software can be resealed.

The metaphor of the bazaar, on the other hand, better describes the Open Source paradigm. Within this paradigm, a single programmer or a group of programmers, in general a group of user with a good knowledge of programming languages, prepare a first version of the software, called development release.

We can identify three major models in which software development projects are managed: the single guru model, the master-disciples model, and the project team model. In the *single guru model*, all the code in the project is written by one single person. She will make the code available for comments, but in the final version, the software code is written by the guru. A good example of the single guru model is the development of the typesetting systems TeX. All the code of TeX is written by Don Knuth. In the *master-disciples model* the initial plan for the software is developed by the master, but code is developed by many people. The decisions about the integration of the software is also taken by the master. The best know example of the master-disciples model is the development of the first release of the Linux Kernel. The initial kernel was developed by Linus Torvalds as part of a project at the university of Helsinki. To get a stable and mature version, he decided to make the source available to the greater community of software developers in the OSS community. During a period of three years software developers of all over the world were invited to contribute, but Torvalds decided to integrate the new pieces of software in the kernel or not. The last model, the *project team model*, or *community based model*, describes the process where the general design of the software, developed by a project team of volunteers, serves as the basis for a large team of people that participate in the development. Contrary to the Master-Disciples model there is a project group that decides what to integrate in the new application, and when to integrate it. Two famous example of successful projects developed using the project model are also two of the most successful pieces of OSS, and probably of software application in general, the Apache Web Server (www.apache.org) and the X package (www.xfree.org).

It is worth noticing how both in the master-disciples model and in the project team model, not the team, nor the leader are rigidly identified *a priori,* or at the beginning of the project. They are fluid and flexible concepts, which can evolve and change during the development process. Developers, who were in the team at the beginning of the project, leave for the most disparate reasons, whilst new developers, with different history and background join[5]. Even if not usual, the leader can change and be substituted[6].

In all three software development models, the software is released in a rudimentary form to the community. The code is not thoroughly tested, as it is supposed to be in the CSS approach, but release in a development version. The community does not expect bug free software, but sees an important role for themselves to detect and to provide feedback to the team, or even patches to overcome the problems. To improve the quality of the software is felt as part of the duty of the community and it is for the community to use, expand and further develop the software. The bigger the community of interested users and developers, the faster the software evolves. At present we see that the Open Source community develops and improves software at a very rapid speed.

2.3 The situation on the market

The OSS initiative is gaining momentum (Lane, 2002). Although Free and Open Source Software has been around since 1970's, the last 10 years we observe an almost exponential growth in the users. In the beginning Open and Free Software was only seen within the academic and corporate research circles in the developed world (Stallman, 1998).

Now, three concomitant effects are playing an important role in the growing diffusion of OSS. First of all, an increasing number of people has a good, and in many cases deep understanding of computers and software, making the more complicated OSS pose less difficulties for individual users. Second, a huge effort has been put by the OS community into simplifying and making more friendly OSS applications. Finally, the number of people, who want or need to personalize the software at hand, is increasing.

We observe a similar trend at the organizational level. An increasing number of organizations are using tools that have been developed publicly. It is not always a deliberate choice. In most cases organizations are just using specific software because it has an important market share and not because it is OSS. A good example of of a successful Open Source Software application that has gained an enormous market share and is therefore used by most organizations world wide is the Apache Web Server. Netcraft's statistics on web servers checks since the middle of the 1990's

[5] The importance for creativity of the continuous change will be discussed in a future paper.

[6] See e.g. Galeon development project (http://galeon.sourceforge.net)

monthly the prevalence of servers on the Internet. The Netcraft research shows that Apache is dominating the public Internet web server market ever since Apache grew into the #1 web server in April 1996. In July 2003, Netcraft counted more than 2.7 Million active websites using Apache, this is 64.0% while the runner up, Microsoft has a market share of 23.8%.

The development of the Linux operating system has also played an important role in the growing adoption of OSS. In 1991 Linus Torvalds, a computer science student at the University of Helsinki, announced on an Internet newsgroup, that he was working on a free version of the well know operating system Unix. Herewith Torvalds followed the same procedure of seeking expertise as Stallman used for the development of Unix compatible kernel Hurd in the Free Software Foundation (Stallman, 1998). Torvalds also declared that he was willing to include in future versions, new features developed by others as long as they would also have been freely redistributable. Herewith Torvalds clearly expressed he adopted the General Public License (GPL) that was introduced by Stallman in the Free Software Foundation (Nuvolari, 2003).

The Linux initiative was one of the most successful OSS projects so far. When version 1.0 was released in 1994, the operating systems proved to be so stable and reliable that could compete with commercially developed versions of Unix. In the second half of the 1990's, Linux was further refined, and the community of developers grew exponentially. The decisive “official recognition” of its potential came in 1999 when an internal memorandum from Microsoft (the Halloween document) leaked out that they considered Linux, and the OSS movement, a major competitive threat for the company. (Nuvolari, 2003).

At present OSS and the Open Source Software development process are considered to be a matured alternative for results that come from the CSS arena. When developing the business case for using OSS in the US army, Kenwood (2001) argues that OSS is a viable long-term solution for the potential for significant costs, reliability and support advantages. OSS is often a good option for products relevant and interesting to a large community with highly skilled software developers. In an open environment with the access to Internet this community is easily available. This understanding is acknowledged by a survey of OpenForum Europe that estimates that almost half of the Chief Information Officers (CIO's) in financial services, retail and the public sector expect to be using OSS in the near future (Wheeler, 2003). when the quality of the software is compared, there are no significant differences in reliability, performance, scalability and security, some argue that OSS outperforms CSS (see Wheeler, 2003 for a detailed description of the different performance variables).

When we consider the commercial success in an historical perspective we can conclude that the OSS initiative migrates into a new stage. In the past five years we see a trend that OSS moves away from the pure developers/programmers domain into a commercial and political arena. The code creators and the early developers are pushed to the background and in stead of the program code, the Return on Investment (RoI) becomes important. The user community is now in the driving seat of the OSS initiative. The user community is not interested in the elegance or the re-usability of the code, but whether the investment in OSS provides some benefits. Benefits are not only financial benefit s, but is they can also consist of increased credibility. Return on Investment needs to be interpreted in the broadest sense, in also includes e.g., governments that emphasize the use of OSS as a way to limit the increasing political power of the multinational business organizations (Hertz, 2001). The current situation is characterized as a chasm between developers community and the user community (see figure 1). This situation potentially dangerous since the developers community, who are the driving force behind the ongoing development of the software, may lose interest and the innovative development of the OSS software may slow down.

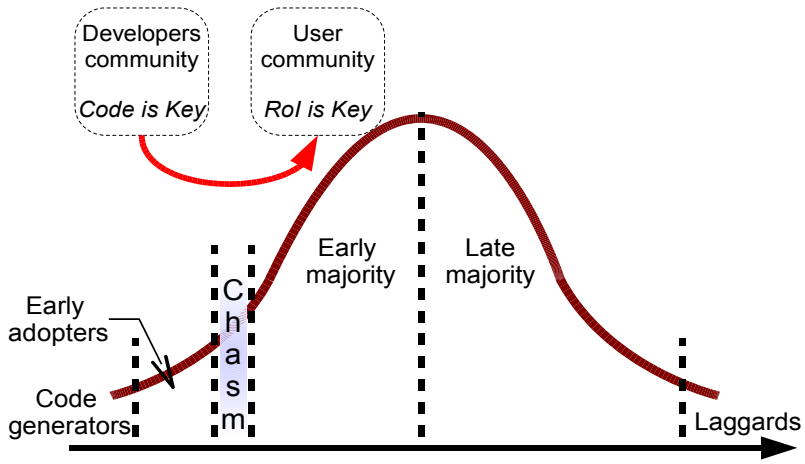

Figure 1: Market Maturity of Open Source Software (Adapted from Briggs & Peck, 2003)

2.4 Advantages and disadvantages of OS

Before we continue with the exploration of the potential of OSS for Africa we conclude this section by identify some of the advantages and disadvantages of OSS. South Africa's National Advisory Council on Innovations summarizes the major benefits of OSS and the adoption of open standards as promoted in the OSS paradigm (NACI January 2002- http://www.naci.org.za/docs/opensource.html):

- Reduced costs and less dependency on imported technology and skills
- Affordable software for individuals, enterprise and government
- Universal access through mass software rollout without costly licensing implications
- Access to government data without barrier of proprietary software and data formats
- Ability to customise software to local languages and cultures
- Lowered barriers to entry for software businesses
- Participation in global network of software development

Additional advantages that are identified the UK Office of Government Commerce (OCG, 2002) are:

- Supplier independence, limiting vendor lock-in
- Patches or updates become available quicker, which limits breakdowns and security risks

At the same time there are also limitations and drawbacks to the use of OSS. The UK Office of Government Commerce identifies the following factors that my limit successful implementation:

- *Available support for OSS*. In the past years support has been lacking a professional approach. In recent years this has improved now that large software companies like IBM, SUN and HP have started to join the OSS movement
- *Finding the appropriate software*: Since OSS is not 'advertised' it can be very difficult to select the appropriate applications for the task it has to support. A more active approach is needed from the users.
- *Documentation*: The documentation that accompanies OSS software application is often idiosyncratic and sometimes non-existent. OSS developers are motivated towards the technical aspects of the application than towards the usability.
- *Limited best practices*: There are very little known and documented cases of large scale migration from CSS to OSS.
- *Hardware – software fit*: OSS often lags behind concerning new hardware. This is caused by the fact the hardware manufacturers fail to release hardware specifications in time to the OSS community.

In spite of the limitations and drawbacks we observe an increasing interest for OSS. At present this interest is mostly within the developed world, but there are aspects that can be of particular interest in the developing world. The most obvious aspect is the cost aspect, for OSS users (individuals and organizations) pay no licensing fee. Cost reduction is increasingly important in Africa, when less donor money becomes available. However, the “openness” and flexibility of Open Source software is more important when considering the question of sustainability in Africa. Open Source software can be customized and constantly revised to develop and change with the needs of the user. And where propriety software is very hardware intensive, Open Source software can be run on computers that are "obsolete" because it has been streamlined and customized accordingly. Let us investigate what is happening in Africa at the moment.

3. Open Source Software in Africa

When we consider Open Source in Africa we have to concentrate on multiple levels in order to get a good understanding of the impact of the different initiatives. The implementation and the propagation of Open Source Software is performed on micro, meso and macro levels. At the micro level we like to think about individual users that decide for or against OSS. At the meso level we consider organizations that take actions to integrate OSS into their total software solution. Finally the macro level were government policies and actions are considered. In this paragraph we will examines some examples of the position of OSS in Africa.

3.1 OSS from a macro perspective

Governments provide a huge potential for OSS, not only as site for implementation for the software, but more importantly as propagators of the philosophy behind the Open Source movement.

Over the past 5 years, a growing number of countries are starting to consider Open Source Software as a serious alternative (www.apc.org). Brazil has been one of the countries that has actively pursued the open source model. It was in Brazil that the first law regarding the use of open source software in the world was passed in March 2000 . Brazil is one of the countries where policies regarding adoption open source software have been successful, notably in the states of Rio Grande do Sul and Pernambuco. Also, the Brazilian Navy has been using OSS since 2002. (http://www.pernambuco.com/tecnologia/arquivo/softlivre1.html)

In Africa, the South African government is one of the forefront player. In the wake of the developments, the South African government released a policy framework document in September 2002 by the open source work group of the Government Information Officers' Council (GITOC) (see for details and discussion about OSS in South Africa: www.oss.gov.za). The GITOC Policy document (GITOC, 2002) recommends that government "explicitly" supports the adoption of open source software as part of its e-government strategy after a comprehensive study of the advantages and pitfalls of OSS for government requirements. Next to adopting OSS software GITOC also recommends that the government promotes the further development of OSS in South Africa. There is an huge potential role for South Africa's SME industry in the production and implementation of OSS as well as in setting up user training infrastructures. At the same time, the OSS approach does represent a powerful opportunity for South African companies to bridge the technological gap, at an acceptable cost.

Some *success factors* need to considered in order to tap this potential:

1. *Implementation should produce value*: Value is related to economic value, i.e., the reduction of costs and saving of foreign currency; and social value, i.e., a wider access to information and computer training.
2. *Adequate capacity to implement, use and maintain*: There need to be enough trained people to support and use the OSS solution. Training users and developers needs to have a high priority.
3. *Policy support for an OSS strategy*: Support for OSS needs to expand to all key players at governmental level, departmental level, IT professionals and computer users in general.

Government's Department of Communication has already begun the move to Open Source by adopting Linux as their operating system. The South African government plans to save 3 billion Rands a year (approximately $338 million USD), increase spending on software that stays in their country, and increase programming skills inside the country. South Africa reports that its small-scale introductions have already saved them 10 million Rands (approximately $1.1 million USD).

Other countries are following. Worldwide, similar moves are discussed by Taiwan, China, Peru, the UK and Germany[7].

3.2 OSS from a meso perspective

[7] Bundesrechnungshof fordert Einsatz von Open Source, 25.02.2002, http://www.heise.de/newsticker/data/anw-25.02.02-004/ (accessed on 2/8/2003)

Recently, the International Institute for Communication and Development (IICD – www.iicd.org), a Dutch Non Governmental Organization (NGO) promoting the use of ICT's in developing countries, investigated the use of OSS in organizations in three countries in Africa: Uganda, Tanzania and Burkina Faso (Bruggink, 2003). The objective of the research was to find out how, where and why organizations from all kind of sectors use OSS, what problems can be observed and what opportunities for development are available. The findings of the research show that OSS in Africa is being used, but it is not yet very wide spread. OSS is mostly found at the server side of Internet Service Providers (ISP's) and is sometimes used by government and educational institutions. This means that Open Source operating systems, mainly Linux and derivatives, web servers, email servers and files servers are found where the day to day computer users to not see them. Large and hierarchical organizations that have migrated completely from CSS to OSS (server side and user side) have not been found. Most of the organizations that are using OSS are small organizations. When the three countries are compared, it is concluded that Tanzanian organizations show most initiatives, while in Burkina Faso organizations do not show interest to move away from CSS.

The research of the IICD highlighted several reasons why organizations in Africa do not take up the challenge of OSS. In the first place there are some false perceptions on OSS. Many organizations believe indeed that OSS is Linux only and that OSS is user unfriendly and only suitable for the ICT specialist. Secondly, there is limited access to OSS. Most of the OSS is distributed through the Internet and with the limited and low bandwidth Internet connections, the access to OSS is limited as a by product. Software companies, including OSS companies, see little market potential in Africa (outside South Africa) and the availability of software is low. This is also reflected in in the amount of resellers for OSS. Finally, there is little expertise available to provide training and support for OSS and eventually consultancy in migration processes.

An interesting example of the introduction of OSS at an organizational level is Uganda Martyrs University in Nkozi (Uganda). In 2002 Uganda Martyrs University embarked on a mission to be the first large organization in the region to completely migrate to OSS. The main reasons for this decision are reduction of licensing costs and capacity building. At present (August 2003) the university has about 250 desktop computers for students and staff, plus a variety of servers, connected in a campus wide Local Area Network. In 2002 the university has migrated the server side (mail servers, Internet connection and file servers) of the network to OSS. In the second phase of the project, starting June 2003, the university will migrate the user side. The university senate

has decided that all standard desktop computers of lecturing staff and students will be equipped with a Linux operating system and Open Office (www.openoffice.org) as a replacement for the popular Microsoft Office suite.

By the end of 2003 the university hopes that 90% of all the computers, servers and desktops, use solely OSS. The project is observed and guided by several OSS communities since it is one of the first integral OSS transformation at both the server and user-side of a large organization. Time will learn the experiences of the project.

3.3 OSS from a micro perspective

Most of the OSS initiatives are small scale projects of individual people or small organizations. A growing number of individuals throughout the African continent is becoming aware of potential of OSS from strategic point of view. Together with relevant advantages from economic and technical point of view, with its lower costs, more flexibility, availability of robust and reliable technology, lower dependence on software vendors, OSS does in fact represent a most important opportunity for changing the position of Africa as whole within the information society.

At user level, and for many individuals in Africa, the challenges of OSS provide new opportunities for development, both at personal and community level. Now that most countries in Africa are connected to the Internet, individual OSS initiatives, which rely on it, are finally thriving. An initiative with good potential that tries to bring together scattered OSS society is the Free and Open Source Foundation for Africa (FOSSFA - http://osfa.allafrica.com/). The initiative started as the offspring of an ICT policy and civil society workshop in Addis Ababa, Ethiopia, in February 2003. During the workshop the participants agreed that OSS is paramount to Africa's progress in the ICT arena. The mission of FOSSFA is therefor to promote the use an implementation of OSS in Africa. Herewith it began to work on a coordinated approach to unite interested individual and to support open source development, distribution and integration. The Free and Open Source Foundation for Africa envisions a future in which governments and the private sector embrace open source software and enlist local experts in adapting and developing appropriate tools, applications and infrastructures for an African technology renaissance. They foresee South-to-South cooperation in which students from Ghana to Egypt and Kenya to Namibia develop programs that are then adopted by software gurus in Nigeria, South Africa and Uganda in order to narrow the digital divide.

On a similar line a number of Internet mailing lists and user groups are emerging, that focus on bringing together OSS developers and users in Africa. At the moment there are active groups working in South Africa, Ghana, Kenya, Zambia, Tanzania, Burkina Faso, and Uganda.

Hosted in South Africa, Internet portals are emerging, that aim at being a starting point for knowledge on OSS in Africa.

Similarly, at commercial level, an interesting initiative has been launched by DireqLearn (www.direclearn.org). DireqLearn promotes OSS and FS as an alternative for the education sector in Africa. By adopting OSS and FS the company can offer new solutions to the educational sector at low costs.

Finally, even if only to limited extend, some African Open Source Software development projects have been launched. Most of the projects are situated in South Africa, for reason connected to the presence of infrastructure. Outside South Africa, a project which is worth of mention is the RULE (Run Up to-date Linux Everywhere - www.rule-project.org/en/) project. Aim of the project is the creation of a very light Linux distribution for people that cannot afford modern computers systems. In order to achieve the goal, developers are modifying a standard Red Hat distribution, trying to allow the greatest real functionality with the smallest consumption of CPU and RAM resources. The new distribution is mainly intended to be for schools and other organizations in developing countries (www.rule-project.org/en/). At the present the RULE project provides a OSS solution with GPL license that is able to transform 5 years old computer models (Pentium 75MHz, 16 MB RAM, 810 MB Hard disk) into useful machines again.

The increasing interest for OSS is also driving the emergence of OSS specific organizations. In several countries of Africa, like Nigeria, Ghana, Uganda and South Africa, specialized software and consulting companies are started, whilst young people with a background in computing are embracing the OSS approach and try to reform the accepted practice of buying CSS software. At present the market share of OSS is still small and difficult for these specialized companies to survive, but when the benefits become clear and OSS is implemented on a bigger scale, the capacity to implement the systems is ready.

4. Conclusion

Over the past years Open Source Software and Free Source Software have matured into a serious alternative when considering new software. The methods and the tools supporting software development processes in distributed environments like Open Source communities on the Internet, have been refined over the past years. As a result software products from the Open Source community have reached levels of reliability and security that allows them to compete with commercially developed software. In turn this gives an important impulse to the growth of the community.

Organizations in the developed world Open Source Software starts to replace an increasingly more wide range of Closed Source Software applications. For a variety of reasons organizations let the advantages of Open Source Software outweigh the advantages of Closed Source Software.

Although most of the implementations of OSS are still on the server side, user side adoption of OSS grows now that friendly environments, high functionality and reliable alternatives for office applications become available. Governments, like Germany, the Netherlands and the United Kingdom, start to promote the use of OSS.[8] Financial and moral support for development and use of OSS alternatives increases awareness and acceptance.

Open Source Software initiatives in Africa are still very limited. When related to figure 1, we can label Africa as early adopters. Except for the South African government, governments in Sub Saharan Africa do not take an explicit position promoting the use of OSS. This is may be partly due to fact that they are not well informed about the possibilities of OSS, but it may also be caused by the fact that these countries have a low level of expertise in the ICT field. At present the skills levels needed for implementing and maintaining OSS are perceived as higher.

The fact that most countries and organization in Africa have limited access to Internet, and when they have access the bandwidth is too low to enable reliable file transfer, does account for a slower pace in the adoption of OSS. Internet has become indeed the premium medium for the OSS community. Traditional means of software distribution (physical outlets) and user support (telephone) are not well developed on the African continent.

[8] EU Observer, Linux conquers Microsoft in Munich, 2003 (http://www.euobserver.com/index.phtml?aid=11435), EU Observer, EU institutions test alternative to Microsoft (http://www.euobserver.com/index.phtml?aid=11261), Volkskrant, 21 augustus 2002, Computerbranch knokt om overheid.

The software development community in Africa is still in its infancy. University programs in software engineering are of relatively recent date, and the quality of the program is low due to lack of facilities, lecturing materials and knowledgeable lecturers (Reijswoud et al., 2003). Training programs in the development of OSS are not in place, which makes that African developers have to rely heavily on the expertise in other parts of the world. High bandwidth Internet access is a precondition for success.

In spite of the low adoption, the Open Source Software paradigm provides advantages that are relevant within the African context. The most obvious advantage is the costing aspect. With increased licensing costs combined with high penalties for illegal use of proprietary software, OSS and FSS provides a low cost alternative. Once the software is acquired, it can be used to automate a whole organization, small or large. Especially in large organization this can lead to a significant cost reduction. A different angle on the costing aspect is the fact that OSS can easily be designed to run on 'obsolete' hardware, like the efforts in the RULE project. The financial situation of many countries in Sub-Saharan Africa does not allow large investments in new and modern hardware. Streamlined software can extend the lifespan of computer hardware without compromising on functionality.

From a capacity development point of view, the openness of the program source code provides the software development community in Africa with an insight on near-commercial software development. African software developers can participate in the world-wide OSS development community and improve their skills from this participation.

From a macro perspective a wide-spread adoption of OSS may provide governments in Africa in the position to negotiate better conditions and improved functionality for the software they acquire. At present governments are the largest buyers of software products in Africa, but they have virtually no influence on the functionality of the products they purchase.

Finally, the flexibility of the OSS makes it the perfect candidate for developing customized applications, which can keep into account peculiarities and specificity of the different local cultures. By adopting the OSS paradigm organizations do not only reduce their costs, but also support a different perspective on intellectual property. If software is 'owned' by everyone, it is also owned by the people on the African continent. This 'ownership' also provides the possibility to influence the

direction of its development, and new, “African” features like the development of user interfaces in local languages, may be proposed.

There is still a long way to go, but the potential benefits are there at the end of the journey. Adoption of the OSS paradigm needs to be encouraged in Africa, as it will represent a significant change in the technological relationship between the North and the South, developed and less developed countries, as we will no longer have to solely rely on the technical expertise of those in the First World. And this represents the first true step towards true sustainability.

Literature

Briggs, J., Peck, M., *QinetiQ Analysis of Open Source Solution Implementation Methodologies QOSSIModo: A Case Study Based Analysis on Behalf of The Office of the Government Commerce*. Report QinetiQ, February 2003.

Bruggink, M., Open Source in Africa: A Global Reality; take it or leave it?. *IICD Research Brief – No UICT01*, May 2003.

DiBona, C., Ockman, S., and Stone, M., (Eds.). *Open Sources: Voices from the Open Source Revolution*. O'Reilly, 1999. Available on: http://www.openresources.com/documents/open-sources/index.html (accessed 20/5/2003).

Government Information Officers' Council. *Using Open Source software in the South African Government: A Proposed Strategy Compiled by the Government Information Technology Officer' Council,* GITOC 2002.

Hertz, N., *The Silent Takeover: Global Capitalism and the Death of Democracy*. Random House, 2001.

Kenwood. C.A., *A Business Case for Open Source Software*. Mitre, 2001.

Lane, D., *A Quick History of Open Source.* Available on: www.open2.org/oshistory.php (accessed on 27/52003).

Nuvolari, A., *Open Source Software Development: Some Historical Perspectives*. Eindhoven Centre for Innovation Studies, 2003 Available on: http://opensource.mit.edu/papers/nuvolari.pdf (accessed 2/8/2003)

Office of Government Commerce (OCG). *Guidance on Implementing Open Source Software*. OCG September 2002.

Raymond, E.S. *The Cathedral and the Bazaar*. 1998 Available on: http://www.openresources.com/documents/cathedral-bazaar/ (accessed 20/5/2003).

Reijswoud, V.E. van, Mbaziira, A., Rollier, B., *Lightning on the Dark Continent, A Survey and Needs Assessment for Information Systems and Computer Science Programs in the East African Region*. Report prepared for the Association of Information Systems, 2003.

Wheeler, D.A., *Why Open Source Software / Free Software (OSS/FS)? Look at the Numbers!*, May 2003. Available on: http://www.dwheeler.com/oss_fs_why.html (accessed 20/5/2003).

About the authors

Professor Victor van Reijswoud (PhD Information Systems) holds the VIA Research Chair at Uganda Martyrs University in Uganda and heads the Information Systems department at the same university. Previously, he was employed a Chief Scientific Officer at Ordina nv in the Netherlands and affiliated as professor at Delft University of Technology. Professor Van Reijswoud has published over 60 refereed academic and professional publications in the area of Information Systems and Business Systems Engineering.

Dr Corrado Topi recently joined the academic world, after a career in private businesses. He teaches Information Systems Strategy and tries to coordinate a new born academic research group on Open Source Software at Huddersfield University in West Yorkshire UK. His main research in the area of information systems strategy, strategic management, change management, ethics of information system, ethics of business management, and, finally, open source software.

Appendix 1 – The Open Source Definition (www.opensource.org)

Origins: Bruce Perens wrote the first draft of this document as "The Debian Free Software Guidelines", and refined it using the comments of the Debian developers in a month-long e-mail conference in June, 1997. He removed the Debian-specific references from the document to create the "Open Source Definition."

The Open Source Definition (version 1.9)

Introduction

Open source doesn't just mean access to the source code. The distribution terms of open-source software must comply with the following criteria:

1. Free Redistribution

The license shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The license shall not require a royalty or other fee for such sale.

2. Source Code

The program must include source code, and must allow distribution in source code as well as compiled form. Where some form of a product is not distributed with source code, there must be a well-publicized means of obtaining the source code for no more than a reasonable reproduction cost preferably, downloading via the Internet without charge. The source code must be the preferred form in which a programmer would modify the program. Deliberately obfuscated source code is not allowed. Intermediate forms such as the output of a preprocessor or translator are not allowed.

3. Derived Works

The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.

4. Integrity of The Author's Source Code

The license may restrict source-code from being distributed in modified form *only* if the license allows the distribution of "patch files" with the source code for the purpose of modifying the program at build time. The license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number from the original software.

5. No Discrimination Against Persons or Groups

The license must not discriminate against any person or group of persons.

6. No Discrimination Against Fields of Endeavor

The license must not restrict anyone from making use of the program in a specific field of endeavor. For example, it may not restrict the program from being used in a business, or from being used for genetic research.

7. Distribution of License

The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.

8. License Must Not Be Specific to a Product

The rights attached to the program must not depend on the program's being part of a particular software distribution. If the program is extracted from that distribution and used or distributed within the terms of the program's license, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the original software distribution.

9. The License Must Not Restrict Other Software

The license must not place restrictions on other software that is distributed along with the licensed software. For example, the license must not insist that all other programs distributed on the same medium must be open-source software.

*10. The License must be technology-neutral

No provision of the license may be predicated on any individual technology or style of interface