Open Source Software Development: Next Steps for the Local Telecommunications Industry

WHM Theunissen, A Boake and DG Kourie
mtheunis@cs.up.ac.za; andrew.boake@up.ac.za; dkourie@cs.up.ac.za
Department of Computer Science
Espresso Research Group
University of Pretoria
Pretoria, 0001, South Africa

***Abstract*— Open Source Software has become a noticeable factor in the Information and Communication Technology arena. This paper investigates this claim and its relevance to the telecommunication industry. A number of misconceptions surrounding open source are explored before introducing the progression from exclusively using open source products to participating in their development. Taking the next step with open source adoption is investigated with the aim of reaping the associated benefits of this development approach. We provide some examples of how these benefits might be gained through some scenarios related to the local telecommunication software development activities.**

***Index Terms*— Open Source Software, Software Development, Telecommunication**

I. Introduction

This article is premised on the claim that *Open Source Software* (OSS) has become ever more influential, and consequently of more importance over the past few years, not only in relation to Information and Communication Technology (ICT) in general, but specifically also in relation to the telecommunication industry.

The remainder of this introductory section is devoted to backing up this claim. Then, in Section II some of the so-called *Myths surrounding OSS* will be identified. Section III then articulates what OSS Development (OSSD) is in relation to OSS. This leads up to a more detailed discussion of it's relevance to the telecommunication industry in Section IV and in turn what the possible next steps should be. Some concluding remarks are provided in Section V.

The importance that the South African government places on OSS in general is reflected in the number of state bodies that have investigated OSS. These include the South African National Advisory Council on Innovation (NACI); Government Information Officers' Council (GITOC); the Presidential International Advisory Council on Information Society and Development (PIAC on ISAD); the Presidential National Commission on Information Society and Development (PNC on ISAD); the National e-Strategy Task Team; the Council for Scientific and Industrial Research (CSIR); and the State Information Technology Agency (SITA).

This research was co-funded through the Centre of Excellence in Teletraffic Engineering for the Information Society (CeTEIS) and the Espresso Research Group, both at the University of Pretoria

Some of the advantages of deploying OSS in Africa in general, and in South Africa in particular, are examined in Theunissen *et.al.* [1]. Some of these benefits are given in the following paragraphs.

The entry barrier into ICT (both use and development) for a developing country tend to be high, especially due to the high costs of proprietary software and high exchange rates. The effect of the licensing cost might be illustrated by a Business Day report stating that it is estimated that the South African government's annual expenditure on proprietary software licence fees accumulates to over 3 billion rand [2] – enough to cover the annual running costs of several tertiary institutions! OSS on the other hand has the potential to help lower the entry barrier due to licensing. This means that OSS can potentially help developing countries to bridge the digital divide and join the information age.

The availability of source code further enables programmers from developing countries to learn from their counterparts situated in the so-called 1st world countries. In turn it also opens up the possibility of contributing to international projects, exposing local capabilities.

Despite this general national interest in OSS, and the clear associated advantages, there does not appear to be significant interest in the South African telecommunications industry. Nevertheless, there is evidence that the international telecommunication industry is becoming alive to the matter. For example, as early as 2001, the European Institute for Research and Strategic Studies in Telecommunications (EURESCOM) produced a report on Open Source[3]. In addition, a more recent EURESCOM investigation has been launched in 2005, known as *project 1552 – Open Source for Next Generation OSS Issues and Challenges Study*[4]. The 2001 report not only confirmed the importance of OSS in the telecommunication sphere, but also predicted the increased relevance of OSS to the industry. That this prediction is being realised, is evidenced by a number of recent developments.

One such development is the continued refinement and development of the Carrier Grade Linux (CGL) initiative. The project was initiated in January 2002 and version 1.1 of the specification was released in October of the same year, with the first distributions passing compliance in 2003. Refinement

and publication of the specification has continued over the years, up to the current release of version 3.2 on the 14th of February 2006 [5]. To date, seven distributions have been listed as compliant with version 2.0.2 of the specification[6].

Another development is the number of Voice over Internet Protocol (VoIP) products that are OSS-based, Asterisk™[7], sipX[8] and Vovida[9] being three cases in point. Asterisk™, in particular, is attracting notable attention. However, these systems are primarily geared towards providing private branch exchange (PBX) solutions.

Furthermore, on the 1st of January 2006 the SCOPE Alliance between Alcatel, Ericsson, Motorola, NEC, Nokia and Siemens was founded. Their stated objective is as follows:

> "SCOPE is an industry alliance committed to accelerating the deployment of carrier grade base platforms for service provider applications. Its mission is to help, enable and promote the availability of open carrier grade base platforms based on Commercial Off The Shelf (COTS) hardware / software and Free Open Source Software (FOSS) building blocks, and to promote interoperability to better serve Service Providers and consumers."[10]

Additionally, Nokia and MySQL AB announced in February 2006 that "Nokia plans to use MySQL®Cluster database technology in its next generation telecommunication subscriber register solution" [11], [12]

As a result of the foregoing, it seems prudent for the local telecommunications industry to carefully consider the next steps to be taken in regard to OSS. In doing so, it is important to be aware of a number of prominent myths that have arisen in regard to OSS.

II. Myths about OSS

As is the case with many technologies, and with ICT in particular, one needs to carefully distinguish between the hype that surrounds the technology and the core of the matter. What follows below is a discussion of a non-exhaustive list of what we believe to be myths surrounding OSS.

All talent must be local. Many proprietary software development organisations believe that good developers are restricted to certain locales, and should be co-located to maximise efficiency/productivity. As a result, development tends to be restricted to specific development 'centres' (also known as 'campuses'), with other branches serving primarily as service and sales-offices. OSS shows that talent from around the world can actually work together on a system without the need for co-location. This suggests that companies should re-evaluate their thinking around co-location in order to utilise talented developers who may not be inclined to relocate to these so-called development 'campuses'.

OSS is completely new. In the early years of computing (from about 1950's to the 1970's) most software was treated as 'open source', in the sense that code and fragments thereof were freely distributed between programmers. This holds true for the original versions of UNIX. The shift to keeping code private gained momentum in the 1970's as companies began to realise the potential of selling their software artifacts. A 'revival' of the sharing culture occurred in the 1980's led, *inter alia* by Stallman and the Free Software Foundation (FSF). In the 1990's, this tendency was further spurred on by the Internet, which enabled ease of distribution across the world. OSS became more mainstream as companies adopted Internet backbone systems that were primarily OSS based, such as Apache; Sendmail; BIND; Linux etc.

OSS is not economically sustainable. The debate here is whether one should give away goods that were manufactured using expensive resources. However, in the OSS paradigm the believe is that one contribute to the common pool because one will be able to 'extract' from that same pool. Thus for example, one might be proficient in developing database software however developing a web-browser is not one's forte. In this scenario one will contribute database software to the community while taking the web-browser for oneself. The aforementioned is one way of thinking on the economics of OSS. Another view is that of giving away the portions of the system that does not generate discernible revenue and only charge for the value-added components and services above it to the markets that require it and is able and willing to pay for it. The growing number of OSS projects that are regarded as economical feasible and lucrative investments further disproves the stated perception. Examples of these projects are RedHat, SUSE, JBoss, MySQL and Eclipse, to name a few. There are a number of business models relating to OSS, most of which are described in literature, including [13].

All software will soon be open source. The need for proprietary software will remain as long as some form of capitalism exists in the world. Certain domains will be driven by proprietary solutions and OSS might not be feasible therein. This is evident in that OSS traditionally seems to be stronger in infrastructure and platform domains. OSS tend to be less mature in the application domains such as accounting, customer relationship management (CRM), enterprise resource planning (ERP) systems. It boils down to the fact that certain domains are better suited for OSS and others for proprietary systems. Thus it would be naive to believe that only one paradigm will reign.

Open sourcing a product results in an increased number of participating developers. Releasing one's product as open source rarely results in a stampede of developers who wish to participate in its continued development. High visibility and interest is required. One first have to establish a market for the product — similar to proprietary software — and to build a community up to a critical mass. This process is not guaranteed to be successful and when it is, it generally takes a long time. The majority of SourceForge projects never achieves this.

OSS always reduces cost and development time. Understanding any software to be used takes time and effort. The learning curve may be increased by factors such as a lack of documentation and/or a high degree of complexity associated with the system. Furthermore, the search and evaluation of available OSS libraries may take up a large amount of time. This also holds true when verifying the alignment of the features provided with those that one actually requires.

OSS is a radical departure from software engineering Feller and Fitzgerald state:

> "On closer inspection, the bazaar model of OSS does not seem to depart as wildly from many of the sensible and proven fundamental software engineering principles as was first assumed. The argument then that OSS begins as a bazaar with a chaotic development process and evolves mysteriously into a coordinated process with an exceptionally high quality end product is perhaps too simplistic a characterisation of what is actually taking place in practice." [14]

This statement indicates that a large number of successful OSS projects are built using well-founded software engineering principles, with the added possibility that non-beneficial and/or bureaucratic practices are excluded. In fact, most of the large-scale and well-known OSS projects adhere to well defined software engineering approaches. One such project is Eclipse, which is primarily driven by IBM—one of the pioneers of software engineering.

Even at the smaller-scale, it is likely that solid software engineering principles are followed. This is evident if one considers that those who engage in OSS project development either do so during working hours as part of their responsibility within an organisation, or, alternatively, that they do so after hours away from their daily job—a job that will invariably involve software development. It is thus likely that those who engage in OSS development will rely on the same approaches and practices to which they are accustomed to in their place of work, unless these are perceived to be unnecessarily cumbersome.

However, in OSS, the voluntary participation and choice of contribution alleviate some common people problems, such as enthusiasm; enjoyment of what they are doing; good at what they are doing. But it may also cause others, for example they are not forced to be rigorous and there is no formal performance management.

Access to source code provides the complete key to software. It might be thought by some that access to source code will allow governments to inspect the security characteristics of code, students to learn from real-world software, and ICT industries to use existing code to develop new products. In fact, it generally takes considerable effort and expertise to assimilate what may amount to millions of lines of code—code that weaves together the results of many different design decisions. Very few people have the skill, tools or inclination to be able to unravel the rationale behind the software merely by looking at the source code. Much more useful is the ability to use or write modular plug-in components that extend a particular platform. This is to be found in a community such as Eclipse, where these components are shared, and the whole provides an extremely useful tool set.

However, depending on the specific project, one might find meta-information surrounding the code base in the form of architectural documentation, mailing-list archives, on-line forums, chat archives, blogs and even the meta-data of the configuration/version control systems. The degree of usefullness and quality of these additional items of information is unique to each particular project.

The above introduced a number of myths that both advocates for and against OSS use. These were provided with the intent to clarify certain misconceptions before introducing the idea progressing to other facets of the OSS paradigm.

III. OSS Development

The first section seems to indicate that *using* OSS should form part of a decision maker's consideration sphere.

Furthermore, a number of misconception surrounding OSS have been identified and was briefly discussed in Section II.

However, merely using OSS solutions forms only part of the OSS paradigm. This may be explained by looking at the four different levels of engagement that are found in OSS. These are:

- *Simply using a product*. One uses some evaluation and/or acquisition approach to select an appropriate OSS product and then use it to satisfy one's particular need. Usually the cost is limited to the time needed to find, download, installing, exploring and evaluating the available solutions. One might also further participate in user forums to gain or share knowledge of the product.
- *Modifying a product without sharing the modifications*. At this level one might discover an additional feature which the product does not provide and furthermore one has the ability to implement the required feature, and does so. However for various reasons one might not contribute this extension back to the original developers.
- *Modifying a product and contributing the changes back to the community*. Here again, one extends the product in some way but in addition contribute these changes back either as a submission to the original project or as an on-line publication. At this level one could even go as far as becoming one of the core developers and/or co-sponsors.
- *Initiating and/or managing an OSS project*. This level of engagement entail that one has either started an OSS based project or one may have decided to release an existing product as OSS. Another possibility is that one might take over management of an existing OSS project due either to prestige gained in the community or by succession of a manager who relinquished control.

Notice that as one progresses through the different levels, greater resource investment is required. This is evident in the fact that downloading and running a package is usually less intensive than hosting and steering a project. Thus, questions to emerge from the above are these: "Why should anyone want to engage at higher levels of OSS participation? Do the benefits of engaging in these various levels of OSS participation outweigh the associated resource expenditure?"

The answers to these questions are highly dependent on the context. However, a number of general benefits are provided in literature (including [15], [13], [1], [16]). An important benefit that we wish to highlight is that of *competency cultivation*. This benefit is related to the adage that "knowledge is power". Participating in an OSS project enables the participants to gain

knowledge of the system due to the openness of both the code and the process. The more one knows about the system the more one is able to maximise its usage as-is, and to extend the system in order to realise opportunities that become apparent.

A number of ICT players have found that following an OSSD approach on certain of their projects leads to a situation where the benefits outweigh the required resource investment. Amongst these companies are Hewlett-Packard (HP), IBM, Sun Microsystems and Novell. HP has even gone as far as introducing and promoting an OSSD approach for internal software development. This, in theory, enables any HP developer from anywhere in the world to particpate on any company project, thus removing the location-based limitation. The aforementioned approach forms part of HP's *Progressive Open Source* concept and is discussed in [17].

The preceding paragraphs briefly shows the potential progression that ICT players might take in regard to OSS and its adoption as a development approach. The next section will endeavours to focus on how this progression can be realised in the telecommunication industry.

IV. The Next Step for the Telecommunication Industry

Section I gave examples of how OSS is beginning to be noticed and indeed used by sections of the international telecommunication industry. However, in Section III the idea was explained of progressing from mere usage of OSS through successive levels of more intensive OSS engagement. In terms of this progression, it would seem that the next step for the telecommunication industry would be to become a *developer* of OSS, rather than a mere OSS user. An organisation may engage in OSSD in two distinct ways.

Firstly the organisation may limit its OSSD to the enhancement of *existing* OSS solutions such as libraries and/or platforms. For example, a telecommunications provider might rely on Asterisk™to supply PBX solutions. However, it could choose to not limit itself to simply installing and configuring the Asterisk™platform for the clients. Instead, it could additionally assign developers to actively participate in the further development of Asterisk™, thereby acquiring a measure of control over future product enhancements.

A second approach would be to adopt a more thoroughgoing in-house OSSD culture. This would be particularly suitable for large organisations with geographically-diverse branches. In this scenario one would maximally leverage human resources, by allowing a set of geographically diverse developers, each with their own niche areas of expertise, to collaborate without the need to be co-located. Thus, for example, Telkom might assign a project to a team of developers spread across its offices in both Pretoria and Cape Town. Another example would be if Vodacom were to constitute a software development team out of developers drawn from each of the African countries in which it is active. Here, the various developers would be able to customise solutions to their local requirements (linguistic, legislative, etc.) and additionally contribute to the common core system. This would simultaneously have the desirable effect of distributing expertise more widely in Africa, rather than restricting growth in expertise to South Africa alone. The same holds true for the Vodacom-Vodafone alliance.

This style of distributed software development corresponds to the first of three levels of strategic open source sharing. These levels have been identified in the POS strategy, defined by HP[17], as: *inner source* (limited—as above—to in-house developers); *corporate source* (where project development incorporates partner organisations) and *open source* (which corresponds to traditional OSS projects). There may well be benefits in taking yet a further step, beyond *inner source* development as described above, to *corporate source* development. In terms of this approach, software solutions are co-developed with partners, suppliers and/or subsidiaries. Additionally it opens up the ability to outsource development to third parties in a more transparent way. This is due to the openness of the OSSD approach and it will be reflected in the interaction between the in-house developers and those of the partnering's developers. An example of corporate source strategy would be if Telkom and Grintek were to work together to develop a solution required by one or both organisations.

V. Conclusion

OSS utilisation in the telecommunication industry is becoming ever more relevant and practical. This holds true not only for software in support of the business side of the industry, but increasingly in regard to the industry's core technological elements. As the adoption of OSS increases, the need to look further at OSS with a focus on maximising the full paradigm in the hope of maintaining an advantage over competitors on the one hand and relishing in the benefits of combined development by partners.

This paper reiterated the importance of OSS leading to the introduction of what lies beyond the usage model, which seems to form the primary focus of advocates. A brief description of potential benefits that may be gained from OSSD was given. A number of scenarios and examples of how this can be achieved was also raised.

It is important that one take a closer look at OSS and OSSD to firstly understand the implications and secondly to exploit the relevant benefits.

However, one should take carefull consideration before adopting OSSD in one's organisation. Successful adoption and the maximisation of the advocated benefits is dependent on the given organisation. In the case of an existing organisational culture that is already collaborative and used to an externally focused view, the adoption of OSSD should be relative easy and well suited. However, if the current culture and continuing paradigm of the organisation tend to be internally focused, then one should preferably steer clear of OSSD.

Further research will be conducted into combining OSSD and Agile Software Development (ASD). ASD is another software development approach that is gaining mainstream adoption. However the ability to find synergy between them may not be as trivial as one might expect.

REFERENCES

[1] W. Theunissen, A. Boake, and D. Kourie, "A preliminary investigation of the impact of open source software on telecommunication software development," in *Proceedings of the Southern African Telecommunication Networks and Applications Conference (SATNAC) 2004*. SATNAC, September 2004.

[2] Information Technology Editor, "State to save billions on software," article Date: 2003/01/20. [Online]. Available: http://www.businessday.co.za/Articles/TarkArticle.aspx?ID=683978. Last Accessed: 2006/08/14

[3] "Aspects of open source in telecommunications: A strategic study on the use of open source in a telecommunications operator's environment," European Institute for Research and Strategic Studies in Telecommunications (EURESCOM), May 2001, project P1044. [Online]. Available: http://www.eurescom.de/~pub-deliverables/P1000-series/P1044/D1/p1044d1.pdf. Last Accessed: 2006/04/12

[4] "Open source for next generation oss issues and challenges study," European Institute for Research and Strategic Studies in Telecommunications (EURESCOM), project P1552. [Online]. Available: http://www.eurescom.de/public/projects/P1500-series/p1552/. Last Accessed: 2006/04/12

[5] Open Source Development Labs (OSDL), "OSDL Carrier-Grade Linux homepage," Website. [Online]. Available: http://www.osdl.org/lab_activities/carrier_grade_linux. Last Accessed: 2006/04/12

[6] ——, "OSDL Carrier-Grade Linux registration," Website. [Online]. Available: http://www.osdl.org/lab_activities/carrier_grade_linux/registration.html/document_view. Last Accessed: 2006/04/12

[7] "Asterisk™homepage," Website. [Online]. Available: http://www.asterisk.org. Last Accessed: 2006/04/12

[8] "SIPfoundry homepage," Website. [Online]. Available: http://www.sipfoundry.org. Last Accessed: 2006/04/12

[9] "Vovida homepage," Website. [Online]. Available: http://www.vovida.org. Last Accessed: 2006/04/12

[10] SCOPE Alliance, "SCOPE Alliance Homepage," Online. [Online]. Available: http://www.scope-alliance.org/index.html. Last Accessed: 2006/04/12

[11] Nokia, "Nokia and mysql collaborate on next generation telecommunication subscriber registers," Online, February 2006, press Release. [Online]. Available: http://press.nokia.com/PR/200602/1034779_5.html. Last Accessed: 2006/04/12

[12] MySQL AB, "Nokia & mysql collaborate on next generation telecommunications," Online, February 2006, press Release. [Online]. Available: http://www.mysql.com/news-and-events/press-release/release_2006_11.html. Last Accessed: 2006/04/12

[13] M. Fink, *The Business and Economics of Linux and Open Source*. Prentice Hall PTR, 2003.

[14] J. Feller and B. Fitzgerald, *Understanding Open Source Software Development*. Pearson Education Limited, 2002.

[15] M. Ruffin and C. Ebert, "Using open source software in product development: A primer," *IEEE Software*, vol. 21, no. 1, pp. 82–86, January/February 2004. [Online]. Available: http://ieeexplore.ieee.org/iel5/52/28149/01259227.pdf?isNumber=28149?=CNF&arnumber=1259227&arNumber=1259227&arSt=+82&ared=+86&arAuthor=+Ruffin%2C+M.%3B++Ebert%2C+C. Last Accessed: 2006/07/20

[16] W. Theunissen, A. Boake, and D. Kourie, "Open source and agile software development in corporates: A contradiction or an opportunity?" Jacquard Conference, Zeist, Holland, February 2005. [Online]. Available: http://espresso.cs.up.ac.za/publications/mtheunissen_etal_jacquard2005_paper.pdf. Last Accessed: 2006/04/12

[17] J. Dinkelacker, P. K. Garg, R. Miller, and D. Nelson, "Progressive open source," Hewlette-Packard Laboratories, Palo Alto, Tech. Rep. HPL-2001-233, September 2001. [Online]. Available: http://www.hpl.hp.com/techreports/2001/HPL-2001-233.html. Last Accessed: 2006/04/12

Morkel Theunissen received a MSc degree in Computer Science in 2004 from the University of Pretoria. He is currently working towards a PhD in Computer Science at the University of Pretoria.

Andrew Boake is a part-time senior lecturer in the Computer Science Department at the University of Pretoria with extensive industry experience. He is also a consultant with SystemicLogic.

Derrick Kourie has been at the Computer Science Department at the University of Pretoria since 1978, and as a full professor since 1988.