Software Development in the Converging Telecommunications World

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
School of Information Technology
University of Pretoria
Pretoria, 0001, South Africa

***Abstract*— Telecommunication technology is increasingly converging towards software-intensive solutions. As the role of software increases, it naturally follows that traditional software development procedures need to be reassessed. In South Africa, this need is especially urgent because of recent legislation that presages an increase in the number of competitors. This paper assesses two important contemporary software development approaches – namely agile and open source – and their relationship to one another. A variant of open source software development that is highlighted, is Hewlett-Packard's so-called 'Progressive Open Source'. This is because it is specifically tailored to corporate use. The benefits and disadvantages of adopting the various development methods are assessed. Guidelines based on project characteristics are provided to indicate when each of the respective approaches should be considered.**

***Index Terms*— Open Source Software, Software Development, Telecommunications, Progressive Open Source, Agile Software Development, Corporate Adoption**

I. Introduction

Recent years have seen a number of changes in the telecommunications industry, both globally but especially with regard to South Africa. Among these changes has been the move towards so-called Next Generation Networks (NGN) and legislation on Voice over Internet Protocol (VoIP). The aforementioned changes have taken place against the backdrop of an increasing convergence between data and voice networks. These changes also form the building block for the current convergence between the so-called fixed-line and wireless networks.

The above transformation highlights the increased importance that software plays within the provisioning of telecommunication services to customers, software being the dominant enabler of these convergences. For example, software has become an integral part of not only the core network (now driven by softswitches, as opposed to hardware based circuit switches of a decade ago) but also of the value-added services (VAS) built on top of the core network. This renewed importance of software seems destined to increase as the future unfolds.

As with the telecommunications industry, the software development industry, too, has seen enormous changes during the past decade. Among these have been the birth of so-called agile software development (ASD) and the ascent of open source software (OSS). As the role of software in the telecommunication industry increases, there is an increased imperative to reassess traditional software development approaches against these more recent trends.

Another trend associated with the South African telecommunication industry is its continued expansion into other African countries. This expansion obviously influences the functionality of the software. Traditionally, this software would have been developed in South Africa. However, the distributed development approaches discussed in this paper could leverage continental-wide regional synergies, resulting in the more effective production of the required software.

The legislation on VoIP has introduced a new competitive forum in which both traditional telecommunications and traditional Internet Service Providers (ISPs) can participate. This development, in conjunction with the growing volume of OSS solutions, needs to be kept in mind during strategic planning.

This brief introduction on the important role that software plays in contemporary telecommunication provisioning—and thus, simultaneously, of the importance of the associated software development processes—lays the basis for the rest of this article. Below, we will introduce two software development approaches that could assist in addressing the current software development needs, not only of the industry in general, but also of the telecommunication industry in particular. These are agile software development (ASD) (Section II) and open source software development (OSSD) (Sections III, IV and V). Section VI highlights the relationship between the two approaches. Section VII brings together the ideas discussed in the preceding sections and provides guidelines for selecting an appropriate development approach. Finally, conclusions are drawn in Section VIII.

II. Agile Software Development

ASD has become increasingly prominent over the past decade. Methodologies such as Extreme Programming (XP) have aroused the interest of a number of software development practitioners and researchers. In fact, even respected institutions such as NASA [2] have begun to explore the use of

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

ASD in selected projects. In reference to Moore's 'Technology Adoption Curve' [12], commentators such as Ambler [3] have concluded that ASD is entering the 3rd phase – i.e. the phase in which early majority adopters (also referred to as the pragmatists) are starting to test ASD on pilot projects.

Unsurprisingly, an increasing number of studies are being undertaken to investigate the strengths and weaknesses of ASD. (See for example [5], [10], [16]) The authors themselves have also undertaken such investigations [21], [18]. In part, these studies involved a theoretical investigation into the suitability of agile approaches for telecommunication software developers. This was backed by an investigation into the extent to which these approaches are actually being used. It was found that certain telecommunication software development teams were indeed already following a number of agile practices. The study highlighted the feasibility of extending the use of these approaches to telecommunication projects of a particular character.

The potential benefits of ASD are: as its name suggest, its *agility* and therefore its capability to adopt to changes (such as business requirements or technology changes); development proceeds at *internet-time*, resulting in faster to-market-cycles; *higher quality* of deliverables when used in suitable contexts. It is clearly important to maximally exploit these benefits wherever appropriate.

The aforemention establishes the premise that ASD should be used to do software development when the project characteristics dictate it to be feasible. In general, ASD is particularly suitable for small- to medium-scale projects, for projects where there is high customer availability during development and where the requirements and technologies may change during the course of development.

III. Open Source Software Development

OSS has matured considerably since its early days of the 1960s. It has withstood the commercialisation years of the 1970s and 1980s when companies tried to eliminate the code sharing culture. It went through the ideological revolution of the 1980s, spearheaded by Stallman and his 'Free Software Foundation'. The 1990s saw the explosion of the OSS culture, driven by the birth and subsequent expansion of Linux, an open source operating system developed by a global community of software developers. This expansion of the OSS culture, developer community, and user community was supported by the Internet. It paved the way for corporate players to explore these solutions and embrace the accompanying culture. Examples are Netscape, who released their browser's source code as project Mozilla [6], [11] and IBM who adopted Apache as their Web Server solution in preference to their own Domino product [11]. This trend increased in the new millennium with Novell's acquisition of SUSE and Ximian; IBM embarking on their open source strategy for Eclipse and Sun Microsystems announcing OpenSolaris^{TM}. Even Microsoft has felt obliged to partially open up their .Net source (shared source) in the limited form of the Rotor project.

These events have resulted in the *European Institute for Research and Strategic Studies in Telecommunications* (EURESCOM) initiating an investigation into the effect of OSS on the telecommunication industry [8]. The report confirms the importance and potential impact of OSS on the telecommunications industry.

In more recent times, a number of telecommunications specific OSS solutions have emerged, such as Carrier Grade Linux (CGL) [13], VOCAL [22], Asterisk^{TM}[4] and sipX [17]. These projects confirm the EURESCOM report. They add yet another factor to the decision-making sphere of telecommunication managers: whether to incorporate OSS products into their software portfolio. Indeed, managers increasingly have to determine whether it makes business sense for targeted products to be developed as OSS.

The authors have also studied the potential impact that OSS could have on the telecommunication industry [19]. Preliminary findings confirm that telecommunications companies need to gain an understanding of OSS and need to recognise its potential benefits and drawbacks.

OSS solutions are starting to impact on the corporate environment. Increasingly, the culture and the development approach behind OSS is attracting the attention of companies. A prime example is Hewlett-Packard (HP) that created and now uses the *Progressive Open Source* (POS) strategy [7]. This approach involves classifying projects into three categories in terms of the extent to which they will be 'opened'. The three categories are *Inner Source*, *Corporate Source* and *Open Source*. The last category refers to the normal full-blown OSS project and thus needs no further explanation here. Inner Source, on the other hand, entails that only the company's own developers have access to the project and only they can participate in its development. However the development approach is still the same as would be used by a general OSS project. Furthermore, similar tools associated with OSS development are used. Following this approach allows the company to maximise the utilisation of it's global developer base and, hence, their associated distributed specialisation and knowledge base. The remaining category, namely Corporate Source, differs from Inner Source in that participation is extended to selected business partners of the company.

One may therefore classify projects undertaken by a company in terms of these categories. Another direction of OSS project classification relates to the different levels of OSS engagement. These levels are briefly described below, starting off with the least involvement and progressing to the management of an entire OSS project. Although certain benefits are associated with each level, each level also requires certain resource commitments, as will be indicated in the discussion.

- *Simply using a product*. At this level one acquires the OSS product in either source or binary form and uses it to fulfil a need. In most cases this implies a negligible expenditure of resources. As a matter of courtesy, one may register as a user of the product, but this is merely to indicate one's support for the product. Another means of expressing support for the product/project might be to donate money to help fund further development.
  The primary benefit of engaging OSS at this level, is the low financial investment required to acquire software to address one's needs.

- *Modifying a product without sharing the modifications*. At this level of OSS engagement, one may decide to customise an OSS product to suite one's specific needs. For a variety of reasons it might be preferable to keep these changes internally instead of sharing them with the community. For example, it might be that the modifications include royalty and/or patent regulated elements, or the changes might represent a significant competitive advantage.

  At this level, the degree of resource investment increases in proportion to the extensiveness of the modifications to the original OSS product.
- *Modifying a product and contributing the changes back into the community*. Here again, one acquires the product and makes changes to suite one's particular needs. However, arrangements are then made to integrate these changes back into the original project or to make the changes available to the community in some other way. The resource expenditure at this level of OSS engagement will also vary in proportion to the extent of the contribution.
- *Initiating and/or managing an OSS project*. Usually, at this level, significant resources will have to be invested into an OSS project. Participation at this level usually becomes necessary when no one else is willing or able to address a need and/or when one is the leader in the project's solution domain. The most noteworthy benefit is the ability to steer (at least to some extent) the direction of the project.

The above paragraphs have briefly introduced the notion of OSS and the role that OSSD can play within a corporate environment. This idea was expanded on by introducing the POS approach created by HP. The following sections will briefly identify potential benefits (Section IV) and drawbacks (Section V) to South African telecommunication companies in adopting an OSS approach to software development.

IV. Potential OSSD/POS Benefits

The following section identifies a number of potential benefits to be gained by telecommunications players in adopting a POS strategy. Where relevant, reference is made to benefits that would accrue to South African players in particular.

Competency cultivation (or knowledge transfer). One of the primary benefits of any OSS strategy is the potential to learn — observers are able to gain knowledge from the inner workings of the solution. This is due to the openness both of the code and of the development process. Having in-house competency of the inner workings of solutions removes the risk and dependency associated with purchasing 'black box' solutions. To have competency in using black-boxed components can take one only so far before the need to have access to the inside becomes crucial. However the competency benefit flows both ways. From the solution provider's perspective, having the customer as part of the development team can be extremely beneficial. It enables the provider to more accurately address the real needs of customers. This is similar to the agile practice of an 'on-site customer'.

Bootstrap IT competency in Africa. With the expansion of South African companies into the rest of Africa, the knowledge transfer effected by opening up source code to foreign OSS participants could have many positive spinoffs. An obvious spinoff is that it would result in an overall increase in the IT competence on the continent. Another benefit would be that by relying on the ability of the foreign developers to identify and capitalise on unique opportunities native to their local environment, solutions could be customised to match their foreign conditions, leading to greater acceptance by foreign users. Yet another spinoff would be to use this joint development as a 'bargaining chip' at the negotiation table—similar to the defence provisioning contracts that were negotiated by the South African government.

A secondary benefit flowing from 'competency building' is the enhancement of *public relations*. Supporting communities and the practice of *giving* rather than taking is likely to be seen by consumers as a positive company practice, leading to greater support.

Sharing of resource investment. By partnering with all development stakeholders, cost and risks associated with projects can be shared, while fostering in-house competency. In addition, collaboration leads to greater access to development resources (such as developers, skills, knowledge and finances) than would otherwise have been available if only one stakeholder had been assigned to do the development.

Another *public relations* benefit and opportunity may be gained by releasing non-competitive and/or incidental software products. These could include drivers for hardware, setup utilities (such as linux setup utilities for ADSL connection) and software libraries (such as XML parsers). This allows the company to 'reduce its commitment to finished projects, leaving it to the OSS community drive the maintenance instead. This could lead to greater return-on-investment (ROI) in the long term, by sharing the maintenance burden with the community.

V. Potential OSSD/POS Drawbacks

Even though OSSD offers a number of important benefits, its potential drawbacks should also be considered. This section will briefly introduce some of these drawbacks. These should be considered by managers who are thinking about adopting an OSSD strategy.

Intellectual Property (IP). IP could be an important concern for corporates who are considering adopting OSSD. After all, the essence of the approach is to expose source code to all participants. This issue, and in particular, its implications in regard to software patents, are continuously under debate by many stakeholders – both in the pro-OSS and pro-proprietary camps. The benefits of patenting software and its consequent retarding effect on innovation is beyond the scope of this discussion.

Cultural change. When a company adopts an OSSD approach, its software developers may have to change even more than when adopting other approaches such as ASD (which, unlike OSSD, is specifically geared towards the corporate environment). Adopting OSSD is a total paradigm and cultural shift for both developers, managers and the company as a whole. To succeed in an OSSD approach one needs to cultivate the disposition that “it is better to give than to receive”. Developers and the team itself need to realise that whatever they do affects not only themselves, but also a whole community outside their sphere of control. Generalising and working as part of a community may thus introduce extra and unfamiliar burdens on the company. The *‘dictatorial’* development approach to which most companies are accustomed is therefore lost in a community approach such as OSSD. Additionally, as is the case with most community based efforts, there is a potential for bureaucratic baggage which could reduce the **agility** of the project.

As stated before, the *cost* associated with hosting an OSS (even more so within POS) may be significant, even outweighing the potential savings that OSS offers. Being the sponsor of an OSS project entails investing hardware, developers, time and administrative resources.

VI. Agile or OSSD?

Suppose that, having considered the benefits (in Section IV) and disadvantages (in Section V) of adopting OSSD (in combination with POS approach), the conclusion is reached that the former outweighs the latter. One would then still need to consider another factor—that of ASD (Section II). The question thus arises of whether to use OSSD or ASD. The authors have explored the possibility of *combining* the two approaches rather than treating them as mutually exclusive choices [20]. The question of compatibility between these two approaches was considered and the results of preliminary investigations are reported. In summary, the report indicates that even though it might seem that ASD and OSSD share similarities on the surface, deeper investigation reveals distinct and contradictory characteristics. The study has refuted the notion that OSSD is just another instance of ASD as suggested in [15]. Its conclusions were arrived at by evaluating the extent to which the agile principles (listed in the agile manifesto [1]) are manifested in the generalised OSSD approach used by the prominent OSS projects and described in literature [9], [14].

The differences between ASD and OSSD are rooted in the opposing development philosophies of distributed *vs* centralised development. This is primarily manifested in the distinct communication models used by each approach. In the ASD case, face-to-face communication is mandatory and the rest of the process is built around this fact; in the OSSD case, remote location is assumed *a priori* and all processes are built upon this assumption.

The study goes on to indicate that the approaches are not entirely incompatible and that it may be possible to reap the benefits that both offer. However, in order to do so, certain compromises need to be made. It was suggested that various development tools would facilitate the changes required. Details may be found in [20].

Emerging from this hybrid development process is the concept of a heterogeneous development model. Figure 1 provides a visual representation of this idea. It shows three distinct types of development approaches involved in the hypothetical project. These are:

- the individual/solo developers (represented by the single circles),
- a traditional hierarchical team (represented by the large circle with a tree structure of individuals),
- an agile team (represented by the large circle with a star communication configuration),
- the core developers of the project are represented by the middle circle, and the project leader is at the centre surrounded by ‘lieutenants’.

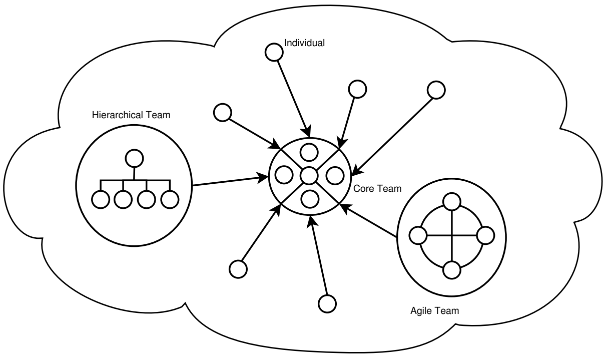

Fig. 1. Heterogeneous development approach

To effectively participate in an OSS project with this sort of configuration, developers need to be aware of the consequences that such a heterogeneous configuration brings to bear on communication at the various levels—i.e. often communication has to occur across new and unfamiliar channels.

VII. A Way Forward

With the convergence of voice and data networks in the past decade, culminating in the VoIP legislation last year, the industry is faced with a high level of competition at various levels. New operators opting for a NGN based solution have a lower capital investment requirement and thus a lower entry barrier to overcome. This is partly due to the fact that telecommunication solutions are moving away from predominantly hardware-based solutions to solutions that are more software-oriented. A case in point is the Eastern Block countries, where outdated networks have been refurbished at a fraction of the cost that would have been required for traditional solutions.

Network operators are not only affected by the software that forms part of the core network. They are also affected by the software (built on top of the core network) that enables the provision of value added services (VAS). As competition between service providers increases, so too will the importance of these VAS offerings.

OSS offers potential cost savings in acquiring solutions for these services and infrastructure. It also has the potential for fostering in-house compentency. Cultivation of in-house compentency is of particular importance when relying on standardised components such as CGL – even if these are only *de facto* standards. Corporates regard it as important to have human resource skills as represented by certified Cisco-, Sun-, Microsoft-, Oracle engineers. Such skills may be described as ‘black box’ knowledge. OSS competency, by contrast, offers ‘white box’ knowledge – i.e. knowledge that enables software engineers to re-engineer components according to local needs. The competitive advantage of possessing such white box knowledge is obvious. Companies should undoubtedly value it more highly than merely having access to black box knowledge.

The implication of the aforementioned is that management is faced with critical decisions as a result of the emergence of OSSD and ASD as potential software development approaches. Prudent management demands that they be actively evaluated in terms of local needs. To ignore them and, by default, continue with existing processes, runs the risk of loosing out to competitors. The foregoing have shown that OSSD and ASD have both benefits and disadvantages. The question is whether the company is willing and able to leverage the benefits and avoid the disadvantages, or whether it will simply leave the field open to its competitors?

The following paragraphs offers a number of *guidelines* for management to consider in deciding about incorporating OSSD and/or ASD into existing processes.

- For projects requiring fast-to-market solutions, based on murky requirements and/or unknown technologies, an agile approach should be considered.
- Projects that do not provide commercial artifacts, could be conducted as full OSS projects. Projects that involve the development of commodity products and infrastructure services such as operating systems, databases, web servers and hardware drivers, are also potential full OSS candidates.
- Partnering and/or outsourcing the development activity is the traditional route that is followed when projects require skills and/or resources beyond those that are available in-house. However, under the new paradigm being propagated here, a corporate source approach should be considered – i.e. positive steps should be taken to more actively involve the in-house resources in all phases of the development, thereby leveraging the aforementioned OSSD/POS benefits.
- If the product being developed is regarded as crucial and/or is seen as offering a competitive advantage, the inner source approach should be considered. By opening up development opportunities on a company-wide basis, (as opposed to the traditional team-based assignment) internal resources are maximally used, while simultaneously protecting the intellectual property of the company.

VIII. Conclusion

We have argued, not only that software has become an integral part of the telecommunication industry, but also that, as a consequence, the need to understand the software development process has become more urgent. This paper has presented two increasingly popular approaches to software development that should receive attention – namely ASD and OSSD. The importance of both approaches was highlighted, with an emphasis on OSS and the development approach behind it. The potential benefits and drawbacks of adopting POS as a corporate OSS development strategy have been discussed.

It has been shown that while OSSD and ASD may seem similar, they are philosophically and practically quite different. However, there nevertheless appear to be synergistic possibilities in merging aspects of the two approaches. Exploring these synergies in greater depth is part of our future research agenda, where an emphasis will be placed on the use of tools to facilitate the merging of the two approaches.

The case has been made that the telecommunication industry should urgently investigate OSS and its impact on future development strategies. The case is based on the increased reliance on software, due to the convergence of technologies in the industry. This new emphasis on software means that, to be and stay competitive, engagement with the new wave of software development methodologies such as OSSD and ASD is called for.

Notwithstanding the need for such engagement, the case that has been made should not be misconstrued as a call for summary universally adoption of ASD and/or OSSD. Clearly, the suitability of these approaches should be judged on a case by case basis.

References

[1] “Manifesto for Agile Software Development.” [Online]. Available: http://www.agilemanifesto.org

[2] N. M. Alexandrov, H. L. Atkins, K. L. Bibb, R. T. Biedron, M. H. Carpenter, P. A. Gnoffo, D. P. Hammond, W. T. Jones, W. L. Kleb, E. M. Lee-Rausch, E. J. Nielsen, M. A. Park, V. V. Raman, T. W. Roberts, J. L. Thomas, V. N. Vatsa, S. A. Viken, J. A. White, and W. A. Wood, “Team software development for aerothermodynamic and aerodynamic analysis and design,” NASA Langley Research Center, Langley Research Center, Hampton, Virginia, Tech. Rep. TM-2003-212421, November 2003. [Online]. Available: http://techreports.larc.nasa.gov/ltrs/PDF/2003/tm/NASA-2003-tm212421.pdf

[3] S. W. Ambler, “Software development magazine’s agile modeling newsletter,” April 2005.

[4] “Asterisk$^{\text{TM}}$ Homepage.” [Online]. Available: http://www.asterisk.org

[5] R. Baskerville, B. Ramesh, L. Levine, J. Pries-Heje, and S. Slaughter, “Is internet-speed software development different?” *IEEE Software*, November/December 2003.

[6] M. Cusumano and D. Yoffie, *Competing on Internet Time: Lessons from Netscape and its battle with Microsoft*. Touchstone, 2000.

[7] 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

[8] “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

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

[10] J. Highsmith, “Does agility work?” *Software Development Magazine*, June 2002.

[11] G. Moody, *Rebel Code: Linux and the Open Source Revolution*. Penguin Books, 2002.

[12] G. Moore, *Crossing the Chasm*, revised ed. HarperBusiness, August 2002.

[13] Open Source Development Labs (OSDL), “OSDL Carrier-Grade Linux Homepage.” [Online]. Available: http://www.osdl.org/lab_activities/carrier_grade_linux

[14] E. S. Raymond, “The cathedral and the bazaar,” *First Monday*, March 1998. [Online]. Available: http://www.firstmonday.org/issues/issue3_3/raymond/index.html

[15] ——, “Discovering the obvious: Hacking and refactoring,” June 2003, weblog entry. [Online]. Available: http://www.artima.com/weblogs/viewpost.jsp?thread=5342

[16] B. Rumpe and A. Schöder, “Quantitive survey on extreme programming projects,” in *3rd International Conference on Extreme Programming and Flexible Processes in Software Engineering*, 2002, pp. 95–100.

[17] “SIPfoundry Homepage.” [Online]. Available: http://www.sipfoundry.org

[18] W. Theunissen, “A case-study based assessment of agile software development,” Master’s thesis, University of Pretoria, 2003. [Online]. Available: http://upetd.up.ac.za/thesis/available/etd-07152004-084708/

[19] 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*. Southern African Telecommunication Networks and Applications Conference, September 2004.

[20] ——, “Open source and agile software development in corporates: A contradiction or an opportunity?” Jacquard Conference, Zeist, Holland, February 2005. [Online]. Available: http://www.jacquard.nl/downloads/JACQ2005/j2005-03-al1530-theunissen.pdf

[21] W. Theunissen and D. Kourie, “A case study of software development processes in the telecommunication industry,” in *Proceedings of the Southern African Telecommunication Networks and Applications Conference (SATNAC) 2003*. Fancourt, Southern Cape, South Africa: SATNAC, September 2003.

[22] “Vovida Homepage.” [Online]. Available: http://www.vovida.org

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 senior lecturer in the Computer Science Department at the University of Pretoria. He has extensive industry experience.

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