Daniel Kofi Calpy

Open source software as a service: Provision of Open source software components to web application developers

Master's Thesis in Mobile Technology and Business

May 14, 2010

UNIVERSITY OF JYVASKYLA

DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION SYSTEMS

Jyväskylä

Author: Daniel Kofi Calpy
Contact information: dacalpy@jyu.fi

Supervisors: Jukka Heikkilä
Department of Computer Science and Information Systems
University of Jyväskylä

Anssi Rantanen
HiQ Softplan Ltd
Espoo Finland

Reviewers: Jukka Heikkilä
Department of Computer Science and Information Systems
University of Jyväskylä

Anssi Rantanen
HiQ Softplan Ltd
Espoo Finland

Title: Open source software as a service: Provision of Open source software components to web application developers

Project: Master's Thesis in Mobile Technology and Business
Page count: 150

Abstract

Over different periods, open source software and Cloud computing have both been described as the next level of evolution in the software industry. Proponents of both paradigms have investigated different approaches to develop, use and create viable business models around computer software. The advantages offered by open source software and cloud computing are often contrasted against the more traditional approaches to software development and product consumption. However, very few studies have attempted to investigate the possibility of merging the attributes of open source software and cloud computing in order to achieve operational and business efficiency.

Previous researches on open source software have identified some problems related to its adoption by potential users. In most cases, open source software has been regarded as a concept which should be left for IT savvy users. Although it boasts of considerable success at the back end and system level, open source software applications have not been very friendly to the non technical end user while problems related to intellectual property rights issues (copyright and patents), software support and maintenance have been identified as major inhibiting factors. Nonetheless, the benefits of open source software cannot be overlooked. In a bid to harness the benefits of open source software and mitigate some of its associated disadvantages, this research seeks for a method of circumventing the challenges related to IPR and support while providing and using open source software to provide an integrated service to a target market of high end users (application developers) .

The purpose of this thesis is to find out how open source software can be offered as part of a service mix for web application developers, identify the characteristics of the primary adopters of the service and the legal, technical and economic constraints which limit its adoption and implementation. This study proposes a generic business model framework for the provision of web application development platform which leverages the strengths of open source software while mitigating some of its drawbacks and targeting a more sophisticated user segment. Increased competition, a need to produce a user centric application at competitive costs, a strong desire to reduce development cycles, tap unto a wide pool of human resource and reach a global market makes it necessary to investigate the idea of leveraging platform as a service and open source software to create new business models.

Keywords: Open Source Software, inhibitors, Back end software, High-end user, adoption, Innovation, Business Models, cloud computing services.

Glossary

AGPL	Affero general public licence
API	Application Programming Interface
ASF	Apache software foundation
ASP	Application service provider
AT&T	American Telephone & Telegraph Company
BSD	Berkeley system distribution
Capex	Capital expenditure
CCIF	Cloud computing interoperability forum
CRM	Customer relationship management
DaaS	Data as a service
DARPA	Defence Advanced Research Projects Agency
DOI	Diffusion of innovation
ECBC	Enterprise Cloud Buyers Council
EPO	European patent office
EPSRC	Engineering and Physical Sciences Research Council
ERP	Enterprise resource process
FLOSS	Free/libre/open source software
Free/OSS	Free and/or open source software
FSF	Free software foundation
GNOME	GNU Network Object Model Environment
GPL	General public licence
GUI	Graphical user interface
HaaS	Hardware as a service
HTTP	Hyper text transfer protocol

IaaS	Infrastructure as a service
ICT	Information and communication technology
IDC	International Data Corporation
IDE	Integrated development environment
IPR	Intellectual property right
ISV	Independent software vendor
IT	Information technology
KDE	K Desktop Environment
LGPL	Lesser general public licence
MPL	Mozilla public licence
NIST	National Institute of Standards and Technology
OCSI	Open cloud Standards Incubator
OMII-UK	Open Middleware Infrastructure Institute – United Kingdom
OPaaS	Open platform as a service
Opex	Operational expenditure
OSS	Open source software
OSSaaS	Open source software as a service
PaaS	Platform as a service
QoS	Quality of Service
SaaS	Software as a service
SLA	Service level agreement
SME	Small and medium size enterprise
SOA	Service oriented architecture
TCO	Total cost of ownership

Acknowledgements

During the course of this work, I have had assistance and support from many people. First and foremost, I wish to express my profound gratitude to Professor Jukka Heikkilä for his highly proactive supervision of the research, decisive guidance, great attention and care, insightful comments, and extensive assistance. He was always ready to respond to my requests, create time for meetings with me whether via email, teleconference, or face to face. His reassuring approach permitted me to achieve the objectives which I set out for.

I am also highly indebted to Mr Anssi Rantanen who always took time out of his very busy schedule to provide assistance and direction. His experience and opinions provided critical insight on the research from a professional point of view. He initiated and assisted in numerous meetings both physically and through teleconference. His professional connections permitted access to some expert personals who acted as interview subjects.

I want to express special gratitude to Dr Anicet Yalaho, Che Abongwa and Mr Tafor Prince Che for their guidance and support during the write up. Their encouragement and support enabled me to come up with a paper which expresses my ideas fluidly, appropriately and in style, while conforming to the conventions of the faculty of information technology at the University of Jyväskylä.

I also recognise the invaluable support of my employer Mrs Monika Poldkivi who permitted me to have a more flexible work schedule so that I could always find the time to attend meetings and carry out the research. She also gave me the opportunity me to hone a much needed skill in multi-tasking and enabled me to see what professional life is like in a truly multicultural environment.

I wish to thank my classmate and friend Fedor Chenogorov for all the exchange of valuable tips and the little chats we had which served to build assurance and opened gateways for new ideas during periods when we met impasses.

I wish to thank all the interviewees and survey respondents for their valuable responses and comments for without them, this study would not have seen the light of day. I also wish to thank all my friends and classmates for their support, criticisms and encouragements.

Finally, I would like to thank the Faculty of Information Technology, especially the Mobile Technology and Business master programme coordinators and creators for giving me the opportunity to study at University of Jyväskylä and conduct the research. I am grateful to all the people of Faculty of Information Technology who either directly or indirectly enabled me to carry out this research.

Daniel K. Calpy

Jyväskylä

2010

Table of Contents

Chapter One **12**

1. Introduction **12**

1.1 Background 14
1.2 Previous research 18
1.3 Current situation of open source software 19
1.3.1 Current open source business models 21
1.3.2 Current challenges facing open source software 23
1.4 Research problem and questions 26
1.5 Scope of the research 27
1.6 Structure of the thesis 28

Chapter Two **29**

2. Literature Review **29**

2.1 History of open source software 29
2.1.1 Free / open source software. 31
2.2 Motivation to contribute to OSS project 32
2.3 Evolution of open source software 34
2.4 Open source adoption and diffusion 40
2.5 Open source business models 41
2.6 Open source software IPR – patents and copyright licence 55
2.6.1 Software patents 55
2.6.2 Open source licences 57
2.6.3 IPR – Patents and copyrights 59
2.6.4 Legal Issues with open source software license 60
2.7 OSS and cloud computing services 65
2.7.1 Functional aspects of cloud computing (service models) 66
2.7.2 OSS in the clouds 70
2.7.3 Cloud challenges 72
2.8 Summary 72

Chapter Three **74**

3. Diffusion of Innovation Theory **74**

3.1 Micro level factors 76
3.2 Macro level (Systemic Change) theory 77
3.3 Organisational level factors 77
3.4 Conclusion 78

Chapter Four **79**

4. Conceptual Framework **79**

4.1 Definition of OSS on PaaS business model 81
4.2 Strategic relationship with suppliers and partners 83
4.3 Value added process 83
4.3.1 Context design 85
4.4 Value proposition 87
4.5 Distribution channel 89

4.5.1 Defining target market 90
4.6 Customer relationship 92
4.7 Revenue streams and pricing strategy 93
4.9 Summary 94
Chapter Five 96

5. Research Methodology 96

5.1 Logic behind the choice of research method 96
5.2 Selection of the research method and data collection 97
5.2.1 Selection of interviewees 98
5.3 Research design 99
5.3.3 Data analysis 100
5.4 The quality of the research 101
5.4.1 Construct validity 102
5.4.2 Content validity 103
5.4.3 Reliability 104
5.5 Limitations of the study 104
5.6 Methods employed to mitigate the limitations of the study 105
5.7 Summary 106
Chapter Six 107

6. Analysis, Implications and Conclusion 107

6.1 Description of respondent groups 108
6.2 Description of survey respondents 108
6.3 Awareness and perception of cloud computing and open source based cloud services 112
6.4 Adoption of cloud platform service and the effect of including open source components on PaaS 114
6.5 Drivers and barriers to PaaS adoption 116
6.5.1 Drivers of PaaS adoption 116
6.5.2 Barriers to PaaS adoption 121
6.6 Conclusions 124
6.7 Major contributions 124
6.8 Implications 126
6.9 Discussions 128
6.9.1 Applications of the framework and benefits 128
6.9.2 Future research directions 128
Bibliography 130

APPENDIX1THE INTERVIEW QUESTIONARE GUIDE 141

APPENDIX 2 ONLINE SURVEY FORMS 145

APPENDIX 3 LIST OF ONLINE FORUMS 150

List of Figures

FIGURE 2.1 EVOLUTION PATTERN OF OSS PROJECTS 37
FIGURE 2.2 EVOLUTION OF COMMERCIAL OPEN SOURCE INITIATIVE 43
FIGURE 2.3MATT ASEY'S FOUR PHASED APPROACH OF DEVELOPING OPEN SOURCE BUSINESS MODEL 50
FIGURE 2.4 LEVARAGING OPEN SOURCE SOFTWARE TO ACHIEVE SOFTWARE PRODUCTION EFFICIENCY 51
FIGURE 2.5 CHART OF SOFTWARE PATENTS BY COUNTRY.......... 57
FIGURE 2.6 A DISAMBIGUATION OF INTELLECTUAL PROPERTY RIGHTS.......... 60
FIGURE 2.7 FUNCTIONAL DIFFERENCES OF OSS LICENSES 64
FIGURE 4.1 PAAS BUSINESS MODEL REPRESENTATION.......... 80
FIGURE 4.2 SWOT ANALYSIS OF OSS PAAS BUSINESS MODEL 82
FIGURE 4.3 PAAS TECHNOLOGY AND SERVICE STACK 84
FIGURE 4.4 ROLE OF VALUE CONTEXT IN DEFINING CUSTOMER SEGMENTS.......... 87
FIGURE 4.5 RELATIONSHIP BETWEEN THE VALUE PROPOSITION AND OTHER ELEMENTS OF THE BUSINESS MODEL 88
FIGURE 4.6 RELATIONSHIP BETWEEN VALUE PROPOSITION AND CUSTOMER SEGMENT 91
FIGURE 6.1 DISTRIBUTION OF RESPONDENTS BY INDUSTRY 109
FIGURE 6.2 REPRESENTATION OF SURVEY RESPONDENTS BY SIZE OF THEIR 110
FIGURE 6.3 RESPONDENTS PREFERENCE FOR DIFFERENT CLOUD COMPUTING SERVICES 110
FIGURE 6.4 PREFERENCE FOR DIFFERENCE CLOUD COMPUTING SERVICE, DISCRIMINATED BY RESPONDENTS' COMPANY SIZES 110
FIGURE 6.5 SIZE OF RESPONDENTS' ORGANISATION BY INDUSTRY.......... 111
FIGURE 6.6 LEVEL OF AWARENESS OF CLOUD COMPUTING 113
FIGURE 6.7 RESPONDENTS' PERCEPTION ON THE INCREASE IN SIGNIFICANCE OF OSS COMPONENTS IN CLOUD SERVICES OVER THE NEXT FIVE YEARS.......... 113
FIGURE 6.8 PREFERENCE FOR DIFFERENT CLOUD PLATFORM SERVICES.......... 114
FIGURE 6.9 PREFERENCE FOR DIFFERENT CLOUD PLATFORM SERVICES, DISCRIMINATED BY SIZE OF RESPONDENTS' COMPANY 114
FIGURE 6.10 INFLUENCE OF OSS ON DECISION TO ADOPT PAAS 116
FIGURE 6.11 DRIVERS OF PAAS ADOPTION 118
FIGURE 6.12 BARRIERS TO PAAS ADOPTION 121

List of Tables

TABLE 2.1 REVENUE LOGIC OF RED HAT, GBDIRECT AND OPENLOGIC 45
TABLE 2.2 DRIVERS OF CLOUD COMPUTING TECHNOLOGY ADOPTION 69
TABLE 2.3 BARRIERS TO ADOPTION OF CLOUD COMPUTING TECHNOLOGY.. 70
TABLE 4.1 CATEGORIES OF CUSTOMER NEEDS... 90

Chapter One

1. Introduction

For over a decade now, one of the strongest buzz in the software industry has been about the unique and maverick phenomenon of open source software. Open source software (OSS) is defined in this study as any software whose characteristics conforms to those specified in the Open Source Initiative website: **http://www.opensource.org/docs/osd).**

Many studies have been conducted about different aspects of this phenomenon and while in the beginning it was just a curiosity to be contemplated mainly by the industry elites and software developers, today it has extended to diverse fields of interests and many business models are being built around the open source concept with Red Hat being among of the most successful. Meanwhile there have been many successful OSS alternatives to some more traditional proprietary software applications such as the openoffice.org, which is quite similar to the Microsoft office suite, NeoOffice, which is an office suite designed for Mac X operating system, Gnucash, TurboCash are alternatives for Microsoft Money(plus), Compiere is an alternative for Quickbooks, Wengophone is an alternative for Skype, Moodle for Blackboard while GIMP and Krita makes a great alternative for Adobe Photo Shop. Some of the most outstanding developments in OSS has not been with creation of end user applications but rather have involved exploits in server side applications, components and technical products such as Apache, Perl, Sendmail, Hibernate, Uportal, Hyperix hq, db4o database engine, Alfresco, Valgrind which are used by technical workers rather than mainstream users.

This research work attempts to analyze the market potential for provision of open source as a service to high end users – This model involves an aggregator organization that interacts closely with both OSS community and developer organizations. It determines the different system components and back-end software required by developer organisations in order to accomplish their core objectives, obtains these components form the OSS community, modify them to meet its customers' desires, host and provide them on a platform, as a service-. The study also examines in greater details, how license and IPR issues affect the provision of such services and the challenges faced by this model. The importance of IPR and post adoption support and maintenance service was identified in a previous research by (Calpy & Chenogorov, 2009)[1] as two of the major factors which inhibit the adoption of open source software.

One main research question arises in this context: How can open source software be offered as part of a service mix for web application developers. To answer this question, three sub questions will be examined;

a) What are the characteristics of the organizations which utilize such services?
b) What types of web application development services are most desirable?
c) How will license type affect the adoption and use of open source software as a service?

In order to investigate these questions, extensive review of relevant published material is done, in addition to face-to-face and phone interviews supported by online surveys.

The results described in this research would assist in the improvement of Open Source Software as a service business model, provide a framework for identifying and categorizing different user groups and provide deeper understanding of business opportunities and challenges related to this model. Furthermore, it will guide both individuals and organizations in deciding what piece of open source software is needed in the industry, where it is needed and how it can best be provided in order to mitigate some of the risks which adopters perceive with its use.

This study examines the business potentials of offering back end open source software as a service to high end users. It attempts to identify which back end OSS are most suitable for this approach of offerings, how the offering should be packaged and delivered and legal issues related to this concept. It also defines specific characteristics of high end users and attempts to explicate why such services are most suitable for them.

In this chapter, the background of the research is introduced. Section 1.1 discusses the significance of this research and the lack of research on this topic. Section 1.2 introduces some previous research in the area of Open source software development, business models, licences and associated cloud computing services. The current situation of adoption and implementation of OSS is discussed in

[1] The author and Chenogorov worked on an initial research project entitled*; Inhibitors to open source software adoption.* From interviews and surveys with respondents, and after perusal of relevant literature, it was concluded that two of the most significant deterrents to open source software adoption where related to IPR and support issues.

section 1.3. The research questions are presented in Section 1.4. The scope of the research is defined in Section 1.5 and the structure of the research is described in Section 1.6.

1.1 Background

Software development and distribution has evolved through a cycle of being open source and free in the 60's and 70's, to becoming proprietary and commercial. With the advent of GNU by Stallman, many companies and individuals rekindled their interested in OSS again. Internet technology has facilitated the development, spread and deployment of OSS. The distribution cost has fallen to almost zero while the scale of coverage is now global. The spread and adoption of Open Source has surpassed all expectations. This can be gleaned from the creator of Linux, (Torvalds, 1999) confession:

> "Linux today has millions of users thousands of developers and a growing market… I'd like to say that I knew this would be happen, that is all part of the plan for world domination. But honestly this has all taken me a bit by surprise…"

Open Source Software (OSS) has radically changed the approach to software development and business strategies related to utilization of IT in companies. The remarkable success of some OSS products has put the phenomenon in the spotlight of many industry stakeholders. Even the staunchest competitors against it are now trying to find ways to form alliances or participate in OSS projects. Microsoft used source code from Berkeley System Distribution (BSD) in its Windows 2000 and XP products while IBM's WebSphere suite contains code from Apache (Krishnamurthy, 2003). Nokia and SUN Microsystems all have strategic alliances with OSS development communities. Some of the more successful alliances are those between Microsoft and Ximian, and Net initiative and Linux. New business models have sprung on every side of the market. Vendors invent new models of making profit considering that the source code is open and therefore by definition, everybody is free to acquire, modify, use and redistribute the software. Clients - individual users and companies - adjust their strategies in order to take advantage of the flexibility which Open source software offers compared to proprietary software. OSS applications in addition to providing freedom in utilization usually have higher quality characteristics and high reliability[2].

[2] Open Source developers aim that as many people as possible access their code making effective the Linux law: *given enough eyeballs all bugs are shallow* (Raymond, 2001)

Today, organizations are not only limiting their Open Source development to web servers running on Apache but are also deploying it on top of their database systems, mobile platforms, e-learning in the public sector, as well as subscriber management in telecoms. It is also widely used for CRM and ERP applications. Some of the biggest concerns about the adoption and use of open source are being dispelled as more companies are of the opinion that OSS has now matured to a point where security fears are no greater an issue than for customers of proprietary software, while many more companies believe that an Open Source strategy offers more value for money certainly in the short term and possibly on the long run. It is also believed that open source provides more stability and gives more control over its management and revision, to the user.

However, for all its promotion, OSS is still being regarded with some suspicion by the majority of the mainstream non-technical end users. These groups of users are pragmatists wishing to use the software as a means to achieve a specific end and not to tinker with the code for the sake of the software itself.

OSS has been described as a knowledge intensive and user centric innovation ((Nemiro, 2002), (Calpy & Chenogorov, 2009), (Lanzara & Morner, 2003), (Krogh; Spaeth;& Lakhani, 2003)). (Von Hippel, 2005) described OSS as an example of a *user centric innovation* were a group of users initiate a project to benefit from using rather than marketing its end product. The public availability of the source code gives the software the characteristics of a *privately provided public good*. OSS possesses two typical attributes of public goods – Non excludability (freely available source code) and non rival (It does not depreciate with the number of users who download it). However, in order to benefit from its unique characteristics, OSS requires the user to be savvy with IT and software development. This demands a large investment of intellectual resources, a situation which may distract the user from their core activities. In addition to this, the OSS landscape looks very chaotic with a plethora of different applications and components been developed and not enough information about the developers, capabilities and reliability of the software. The user is at a loss of knowing what product does what, how reliable the product is, what legal issues constraint their use and who are responsible for the support and perpetuation of the product.

SaaS has been considered as one of the main competitors to OSS. However, the major differences between SaaS and other types of software is not with the technical part of the software itself rather it lies with the manner of hosting and offering (Andreas,Timothy & Bobby, 2006), (Lehman, 2008), (Greg, 2008)). Subscription revenue model is most popular among SaaS providers. A fee is charged on a temporal basis for usage of the software.

Another difference between these two types of software is the cost of implementation. One of the arguments for OSS is that there are no licensing fees compared to proprietary software. Nevertheless, (Andreas;Timothy;& Bobby, 2006) and (Hancheng & ChangQi, 2008) argue that the cost of implementing SaaS is generally lower than for OSS. They hold that even though no license fee may be charged for an OSS product, the total cost of implementation and support for OSS may ultimately be very significant while SaaS on the other hand has a much lower cost because it is a fully hosted solution with low infrastructure and entry cost. In addition to this, SaaS can be deployed rapidly and its support and maintenance provided unobtrusively hence, *enabling the company to concentrate on the main business activities* (typical for any kind of outsourcing). SaaS, however, has its drawbacks, too. Some of the most significant ones are privacy and security. Secret and sensitive commercial information has to be stored off-site – on the providers' servers, which may be vulnerable to crash or hacking. Therefore control over sensitive information would be given to outsourced services. Another problem which is typical for SaaS is vulnerability of the shared multi-tenant architecture. This implies that servers of a SaaS provider may be attacked and brought down. Also, in the busiest hours service may go down causing business outage.

Although there are obvious differences between them, possibilities of symbiosis between OSS and SaaS are, however, not out of the question. With the emergence of cloud computing, the use of OSS as the basis for offering SaaS has been voiced in some quarters of the software industry. Open source software as a service (OSSaaS) has been described as the idea of offering OSS applications or component through a hosted on demand delivery model. At the moment, some companies are considering various possibilities. (Subramanian, 2008), explored how OSS can add value in the SaaS world by building the confidence of customers.

One of the greatest risks in adopting SaaS is the fact that customers put their data on the hands of third party vendors. In the event of termination of business relationship between the vendor and user organisation, the customer may not only have to get their data out of the vendor's server, they also need to find a compatible SaaS application to keep going. Unless the old vendor is offering an option to export the data in an open format, the customer may face big problems. In addition to this, many customers may want to stick with the same application due to various reasons. This is the kind of scenario where releasing the SaaS application as Open Source adds value to the customers. They could just install the application in one of the cloud infrastructures available and keep going. On the side of the vendor, using OSS as a platform for developing SaaS offering may result in a cheaper and more robust product. In this situation, the vendors could realise greater profit margins or make their offerings more competitive by transferring the cost savings they achieve with the use of OSS to customers in the form of reduced service fees.

Using SaaS as the conduit for providing OSS has the advantage that, it removes most of the encumbrances associated with adoption of OSS as it insulates the user from the worries of appropriate license strategy, implementation, support and maintenance. Version control and updates are handled remotely and automatically based on the users' specification, releasing valuable time and intellectual resources for the company to concentrate on core activities while retaining control of the source code. Cloud platform as a service provides the option for developers to develop web application in an integrated software development, testing and hosting environment. Prebuilt business functionalities offered in such platforms enables developers to build and access the applications which they need without the hassles of maintaining whole software system infrastructures.

Although many individuals and organizations have attempted to latch their business and operational concepts on the OSS notion, most have only had a modicum of success (Krishnamurthy, 2003). Most customers find it difficult to break away from established cultures in the IT industry; another factor is that, most OSS projects to not reach stable state as contributors loose the motivation to continue or the need to maintain organization in the community becomes over whelming.

According to (Stefan, Fabio & Maria, 2007), applications towards sophisticated users have a larger probability of evolving in the development status towards a stable release. Although many researchers have observed that end user applications is the most downloaded type of open source software, developers' efforts mainly focus on behind the scenes system applications, programming environments or utilities providing basic functionalities to an operating system (Keiran & Schussman, 2003). In other words, the most notable OSS projects have been those concerned with the development of back end software as opposed to the more trite desktop applications. ((Karim & Robert, 2003), (Hertel, Niedner & Herrmann, 2003)) provide some explanation for this by pointing out that, the two most relevant reasons to contribute to an OS project are related to the desire of being intellectually stimulated and the wish to learn from other OS developers. Both of these goals are most likely achieved by participating and actively contributing to technically more sophisticate and challenging projects. These arguments suggest that sophisticated applications for high end users are more likely to stimulate the contributions from OS developers and, consequently, to succeed. This point is buttressed by (Krishnamurthy, 2003) who asserts that technical products such as Apache tend to be used by technical skilled workers rather than lay employees. While Linux have excelled on the server side software, they have not done so well on the client side. He goes further to state that the mainstream user wants performance and features over reliability, one easy-to-use suite rather than an extended series of options, and probably does not care about access to the source code.

Today, it cannot be denied that OSS has come of age as a reliable way of developing software. A significant amount of research has been done on the subject and many development and business models have been proposed. However, OSS communities lack the resources to market their programs through traditional media. While it is easy to build awareness among developers and technical users, it is a lot more difficult to gain mainstream approval. With the emergence of successful models such as Red Hat, Caldera, Debian and Lycoris, more resources is being invested into the study of OSS market dynamics

1.2 Previous research

Research on free and open source software development projects has so far largely focused on how the major tasks of software development are organized and motivated ((Scacchi, 2004), (Raymond, 2001), (Pavlicek, 2000), (Karim & Robert, 2003), (Hertel Niedner & Herrmann, 2003), (Lerner & Tirole, 2000)). Research has also been done on the different legal issues surrounding the distribution, adoption and use of OSS including issues of intellectual property rights ((Seppä, 2006), (Dixon, 2003), (Ueda, 2005), (Ming-Wei & Ying-Dar, 2001), (Välimäki, 2005)). A good number of researchers have also focused on the different OSS business models and inhibitors to the adoption of OSS, examining their strengths and weaknesses ((Hecker, 1999), (Daffara, 2009), (Riehle, 2007), (Bonaccorsi;Rossi;& Giannangeli, 2004), (Hohensohn & Hang, 2003), (Krishnamurthy, 2003), (Madanmohan & Pal, 2002)). The findings of these studies reveal the motives of OSS developer communities and individuals who participate in them, how such communities are structured and coordinated, the risks and legal issues involved in adopting an OSS business strategy and measures to define success in OSS projects and adoption. Intensive literature review has been done, (Chapter 2) and the outcome suggests that very little research has been done on the possibility of using OSS as a platform for providing SaaS while the author could not find any research work which actually looks into the very intriguing idea of offering back end OSS as a service on a development platform to high end users.

1.3 Current situation of open source software

The close of the first decade of the 21st century has seen open source software establishing its place in the software industry. It has become ubiquitous in the IT world, with wide-spread use in Fortune 500 corporations as well as in universities, developing economies, governments, and student populations (Sun Microsystems,Inc., 2009). Indeed, open source has become the leading approach in many business computing categories. But the most pervasive penetration has been in the area of embedded systems, business servers and nowadays, complex infrastructure platforms - application servers such as JBoss, Zope, JonAs and Enhydra. Open source is also moving into the enterprise applications space. With successful companies like SugarCRM creating enterprise-ready Open Source applications. According to Gartner’s estimates, at least 80% of commercial software will contain open source code by 2012[3]. In contrast, open source software has not yet been as successful on user desktop systems, although Free Software provides a large part of Apple's MacOS today, as well as essential elements of Microsoft Windows. However, the current atmosphere in the software landscape indicates a shift in value from desktop applications to a more distributed architecture.

> “Today we are seeing much of the value of software move from the desktop to the network, an area in which we are already entrenched. This can only lead to the expansion of Open Source on the systems in individual user's hands (Bruce, 2008)

This shift works to the strengths of open source software especially as it appeals to an area generally suited to technical experts. According to (Bruce, 2008), the largest part of the payment for Open Source development today comes from cost-centre budgets of IT users, rather than profit-centres and most open source software today is still being produced by users, for users. In most cases, the profit centres of the developers are not tied to software sales, but to some other business.

[3] Gartner. Gartner Highlights Key Predictions for IT Organisations and Users in 2008 and Beyond. http://www.gartner.com/it/page.jsp?id=593207.

Today, there is an existing trend for governments and start up IT businesses to build elaborate strategies mainly on open source software or a hybrid predominated by OSS. The Brazilian, Argentinean and UK governments have been strong proponents of OSS in recent years. China has cited national security as a reason to develop and adopt a version of Linux called Red Flag[4] while other OSS projects are mostly favoured by universities and other public institution. With the advent of the 2009 global economic crisis, many industrial leaders believe that OSS will see an even bigger surge of adopters[5]. OSS has been recognised as a great way of encouraging innovation and creativity although there are also others who see it as a destructive trend that is threatening to undermine the economic viability of the entire software industry. However, many companies, institutions, and individuals now share innovation on a daily basis, entirely in the open, through Free Software development communities. (Bruce, 2005), points out that this system of public innovation eliminates the high transaction costs of lawyers, lawsuits and licensing. It focuses on building a fertile community across the market for idea creation and utilization rather than dividing the market for the direct monetization of ideas as property.

But the phenomenon of OSS is no longer limited to the OSS communities and users whose primary objective is not to monetize it. Traditional proprietary/ closed software companies, vendors and integrators are all embracing open source technology for software infrastructure and products (Mathew, 2009). The prevailing economic environment also serves to persuade large companies to consider integrating open source technologies and consolidation techniques such as virtualisation into the solutions supporting their business systems and processes.

New models of participation in the open source movement are emerging rapidly. Several companies are increasingly releasing some of their proprietary software systems as open source on the one hand and acquiring open source software on the other hand. Examples of two big companies are Sun

4 The penguin sees Red- CIO: **www.cio.com.au/article/138322/penguin_sees_red?fp=4&fpid=21**

5 According to Matt Asay (Vice president of business development at Alfresco), “as IT budgets dry up, there will be much less inclination to bet on new projects. At least, not those that requires significant capital investment. What we may end up seeing is a lot of dabbling in open source during the recession, preparing to ramp up payments into open source once the economy resumes growth”. This view is also supported by other industry leaders like Dave Rosenberge of mulesource and Javier Soltero of Hyperic.

http://www.infoworld.com/d/developer-world/state-open-source-issues-and-opportunities-416

Microsystems and Nokia. In the former setting, Sun has released the Java Platform Standard Edition for Business to their customers while Nokia has acquired Symbian and is giving it away as open source. In the latter setting, Sun has acquired MySQL and Nokia has opted to use QT (*a software toolbox for developers, which lets those developers port their applications to any of the major operating systems such as Linux, Max OS X, Windows, Windows CE, Embedded Linux and S60*) after acquiring Trolltech. Some companies such as Hyperic with SIGAR, Digium with Asterisk and Sun with Solaris, have both an open source version of their products and an enterprise version which comes with special features and enhanced customer support (Sacks 2009). The benefit of this approach is that the open source community is constantly working on the open source version of the product while in house development team is integrating whatever the company determines to be beneficial to the product.

1.3.1 Current open source business models

(Dave, 2009), examined the current commercial open source business models and notes that, one of the biggest challenges for vendors trying to monetize open-source products is how to encourage payment for open source products or services without reducing the worth of the user base that is hooked on the free software.

(Cote, 2009), grouped the prevailing business models into four very broad categories and asserts that the biggest problem for open source companies to sustain large financial success is that of scalability. The four main groups are:

1. **Support and services:** Support here implies, troubleshooting a customer’s problem using the software and offering solutions to make for better user satisfaction and productivity. It may be offered by phone, email, or even on-site visits. The support contract may be differentiated into classes such as: Gold, Silver, Platinum that scale up the response time, availability (business hours, weekends, 24/7), named contact, and other features that do not directly connect to the software itself. While services are considered to consist of such issues as, integrating and customizing the software to meet customers’ business needs and fit with the legacy system. Examples here include SpikeSource, SourceLabs, and OpenLogic, help enterprises address integration and support issues, putting together common open-source products into stacks and then providing services around them. OpenNMS Group (which maintains and supports OpenNMS® project; an enterprise-grade network management platform developed under the open source model) is another example; all of their software is

open source while their revenue comes from the supply of services and support such as expert consulting services, training, and custom development. The biggest problem with this model is that profit margins tend to be low and most independent software vendors are not very enthusiastic about having to customize their software per customer.

2. **Patches & Updates:** This model looks at the premise that all software (including OSS) is full of bugs, security problems, and other issues that need to be fixed of improved. New features are added to software all the time (new versions in the case of proprietary software). However, in the case of OSS, the additional features come as small incremental releases. The patches and updates open source businesses model aims at selling quick, easy, and even early access to updates. The challenge here is that the underlying code for these updates are also open source thus it is also possible for the customer to just get hold of them, re-compile, and deploy the them. There is certainly a very low entry barrier for competitors in this model coupled with the difficulties in reaching a significant market scale another big question is, how much revenue can any given company depend on for simply providing the service of updating open source software?

3. **SaaS and Cloud:** Another option is for vendors to charge for actually running the software for their customers. They may put the software up in the cloud and charge for the service they are providing instead of the software itself. An example of a company using this model is WordPress. The customer can download WordPress for themselves and run it or they can pay to simply have WordPress.com run it for them. Interestingly, even though the WordPress instance that the customer gets at WordPress.com – that is paid for – has less features than the one they can download for free, The challenges of personally running a blog and elastically scaling up when it get lots of traffic is worth not only paying for, but also sacrificing some features and flexibility. Although closed source SaaS's like SalesForce[6], Netsuite have established themselves as successful business models, taking advantage of the developments in internet technology and the advent of cloud -centralized computing services

[11] SalesForce generates revenue of more than $1 billion a year - a 60% five-year annual growth rate - all from providing software subscriptions to businesses.
http://money.cnn.com/2009/02/16/technology/hempel_salesforce.fortune/?postversion=2009021709

that are delivered over the Internet – computing. The shift towards Open source SaaS business model is still lagging behind even though it seems to have a lot of potential. An important issue to note however is that, often, to run software at large scale as a SaaS some additional codes are required in order to manage it and balance out databases and web servers. The additional code may be licensed as closed source as in the case of Google. In this case, the client will not be able to access the full functional source code if he requests for it.

4. **Aggregate and Package the software; Sell Closed Source:** This is an increasingly popular open source business model. Effectively, the vendor is selling closed source. The closed source software typically comes in two forms: additional software used to manage the open source software or “plugins” and “extensions”. In both cases, typically, the closed software is being sold to non technical end users. Examples of businesses which employ this model are; Zenoss, Groundwork, and Hyperic. In each case, the vendors use open source software components to build and offer closed source software to the customer. In the case of Zenoss, the customer gets additional functionality in the form of “ZenPacks” that allow them to monitor and manage more processes in their data centre for example, VMWare. Groundwork and Hyperic pull together and polish all the open source projects which they amalgamate, and add further functionalities to this, for better user experience and efficiency. Furthermore, Hyperic offers white-labelled versions of their management software to other open source vendors, developers and administrators. For example, when a customer acquires Mule from MuleSource, they also get a version of Hyperic’s HQ tuned to monitor and manage their Mule install.

1.3.2 Current challenges facing open source software

Patent: Amongst the many challenges faced by the software industry in general, and the open source software movement in particular, a software patent has generally been cited as the most critical. (Bruce, 2005) points out that the biggest problem facing open source is software patenting. He claims that although producers of proprietary software may to some extent be protected by copyright, even they are threatened by software patents as evidenced from the number of court cases brought against them. Today, it is very easy to get software patent for something that is not really an invention at all. Perens cited the case of the JMRI (Java Model Railroad Interface)[7] to buttress this point. Perens also cited running law suits against a component of JBoss (Red Hat is the defendant), and against the ClamAV anti-virus software (Barracuda [integrator] is

[12] An Open Source developer's work, a JMRI, was integrated into a commercial product, a model railroad throttle, and then the throttle's manufacturer brought a patent suit against the very Open Source developer whose work he capitalized upon.

defendant) for the "invention" of integration of virus checking into email transfers. The current atmosphere means that developers, integrators and vendors are all expected to be very aware of the types of licences and patents surrounding their products in other to make it effective, and to protect themselves from software patent claims, or spend much money (which most individuals and start ups do not have) on expensive claims and lawyers.

(Fotescu, 2007), highlight the problems of patent today in the open source industry. The multiplicity of patent types whose definitions are not always clear means that individual developers who are usually not the savviest with legal issues and who are not financially or politically equipped enough, become the most vulnerable to claims from stronger, more established organisations. The idea of patenting software he claims, is also counterproductive to the very end to which patents should serve - *The US Constitution states in Article 1: «Clause 8: To promote the progress of science and useful arts, by securing for limited times to authors and inventors the exclusive right to their respective writings and discoveries.»*[8] -. Perhaps the most disturbing issue regarding patent is the ambiguity of interpreting the law as to the applicability of patent, on a proposed innovation. Although certain jurisdictions, such as the US, allow more liberal patenting of software-based inventions, these patents cannot be enforced in the UK[9]

(Fotescu, 2007), also raised another problem associated with the use of open source software; the end user of software is not indemnified in the event of a claim for using the software. Even though he might have purchased or obtained the software from a vendor or distributor, the law assumes that, because the codes are open for him to examine and determine if the software infringes some other vendor’s patent, the end user is just as liable as his supplier for infringement. This aspect of patents makes corporate customers especially; feel unsafe to use open-source software, because as long as the code is widely available, possible patent-infringing code snippets could be identified at any moment.

Business model: It is very difficult to be profitable by selling support services associated with free, open-source software (Daffara 2009, Fotescu 2007). The individual customer is unlikely to be willing to pay, and the corporate penetration of Linux is mediocre, especially in North America, where Red Hat would be the only Linux trusted for large-scale deployments, with a second choice

---

[13]Constitution of the United states of America: http://www.house.gov/house/Constitution/Constitution.html

[9] The United Kingdom's PM had a prompt response to a petition on software patents, in which it was confirmed that *«the Government remains committed to its policy that no patents should exist for inventions which make advances lying solely in the field of software http://www.number10.gov.uk/Page11077*

being Novell. Mandriva is left as the only purely European Linux vendor, but it too is struggling for profitability. According to (Jason, 2008), Javier Soltero – CEO of Hyperic points out that:

> The biggest challenge for open source companies remains finding a scalable way of generating revenue while maintaining the openness and community focus that makes their business possible. Today's economic climate will put even more pressure on business models that are not sustainable because not enough value is being delivered to customers to motivate them to pay.

Many open source based start-ups are finding it difficult to generate huge revenues based entirely on an open source strategy. Most of the current business models are faced with a limited market opportunity, excessive competition due to the relatively low barrier of entry from third parties and small profit margins for support and services.

Source Code and documentation: One of the strongest arguments in favour of the integrity and reliability of open source software is related to the fact that because the source code is available for anybody who wishes to acquire it, it is easier for both users and developers to examine the code discover and fix bugs. With the diversity of available Software Bug Tracking Tools, it has become easier for software bugs to be detected and reported. However, (Fotescu, 2007) argues that although a public bug tracking system leads to a better feedback, and project management, with the side effect of having zillions of bugs reported, most of the very common bugs reported are not fixed, for example the Openoffice bugs described below. It is difficult to find an easy way to change the default paper from Letter to A4 in all the Openoffice.org (Bug #39733)[10], or the design flaw that limits paragraphs to 65,534 characters, as if it were under Windows 3.1 (Bug #17171)[11]. (Fotescu, 2007) claims that although Openoffice.org is open source, nobody is going to diverge it just to fix such annoying bugs. According to him, once a product becomes conventional, it is almost treated like a proprietary product: Everybody is going to accept it and its flaws for the sake of the compatibility thus diminishing the concept of “freedom” in open source world.

Another disturbing issue with having the codes public is that of security. While security fixes are benefiting from having the code in the open, it also exposes weaknesses in the software and hackers do not have to try blind attacks anymore as they can easily identify the precise errors in the software.

[10] http://www.openoffice.org/issues/show_bug.cgi?id=39733
[11] http://www.openoffice.org/issues/show_bug.cgi?id=17171

Documentation is a very important aspect in software engineering but it is still a very challenging issue for many software developers. Many open source projects face significant challenges in generating and maintaining high quality, end-user documentation. (Fotescu, 2007), points out that although Red Hat and Novell issue quality end-user Linux documentation, this is often far from the quality of the old-style UNIX printed manuals. He also claims that KDE and GNOME always ship their products with incomplete and obsolete help files while Linux has gradually lost the classical respect for ***man*** pages -a long-time UNIX tradition-. According to (Yeats, 2008), since open source software is focused on developers, often to the exclusion of other contributors, it loses the benefits of the end users experience in the documentation because the best people to write user documentation are the users themselves. Another aspect he pointed out is the fact that most contributors to open source software projects are searching for exciting activities and unfortunately, in open source projects where contributors have significantly broader freedom over what to contribute, very few of them contribute to anything that does not excite them. In addition to this, the bulk of information generated by open source projects is often in mailing lists, forums, and chat logs. Few projects have mechanisms for integrating the useful information into formal documentation.

1.4 Research problem and questions

The strengths of open source software have been discussed in the previous paragraphs and this has been used in an attempt to explain its growing influence in the software industry. Current trends have also been identified together with some serious challenges. The use of open source software is no longer limited to trivial applications. As more mission critical undertakings are being based on open source strategy and both low budget start-ups and multi-billion Euros incumbent look towards creating economically sound businesses around the open source phenomenon, research is important to identify the characteristics of the target market, the target market potential, the specific type of products and services desired by the chosen market and the most suitable business model which satisfies both the supplier's and the customers' expectations . Literature review reveals that there is lack of systematic research on the supply of complex open source software as a service to high end users. There is no detail framework to guide company executives and managers on what strategies to adopt and where to concentrate their resources.

The research question addressed in this study is: ***How can open source software be offered as part of a service mix for web application developers?***

Prior literature suggests that this research question should be divided into the following specific questions, which are useful in building the skeleton for executing a framework for the provision of back-end open source software as a service to high-end users:

1. What are the characteristics of the organizations which utilize such services?
2. What types of web application development services are most desirable?
3. How will license type affect the adoption and use of open source software as a service?

The study addresses the research question mainly by:

(1) Conducting empirical investigation, which will comprise of: interviewing and consulting with researchers and practitioners.

(2) Conceptualising a business model framework on open source enabled software development platform as a service and review of relevant literature in the fields of cloud computing, open source software adoption and licences.

(3) Conduction of extensive online survey among different cloud computing and open source forums

1.5 Scope of the research

The study deals with the aggregation and supply of back-end open source software - software that performs either the final stage in a process, or a task not apparent to the user - components and applications to developers. It examines the business and legal climate for this endeavour, investigates the characteristics of potential adopter companies, the specific products and components and their suitability of supply as a service. It develops a conceptual business framework of software aggregation and provision on a PaaS and validates it through the investigation in online communities and forums of cloud service providers and users and also online open source communities. Specific software components and applications of interest include content management software, data base engines and middleware applications targeted towards developers who may use them to build and provide end user software offerings.

1.6 Structure of the thesis

Chapter 1 lay out the background of the study, the current situation of open source software, the research problem and questions, the scope of the study and the structure of the thesis.

Chapter 2 reviews the literature concerning open source software development, economics and business models, patents, licences property law and cloud computing.

The literature review in OSS development includes the following issues: Historical perspective of OSS development, major types of open source software, open source software diffusion and evolution. While Open source economics and business model will include an in-depth examination of different open source business models and a critical analysis of the revenue structure and potential of the different models and the rational for companies to pay for open source software services.

Chapter 3 explores the diffusion of innovation theory through a literature review. The review on the diffusion of IT innovations includes micro level factors, telecommunications and IT industry level factors, and macro level factors.

In Chapter 4, a framework for the aggregation and delivery of developer oriented open source software on a development platform as a service is developed and described. The provision of back end open source software as a service to high end users process consists of:

1) strategic analysis and decision, 2) building partnerships with open source communities and cloud platform users and enablers 3) target market research and promotion, 4) aggregation and packaging of open source components, 5) Investigation of delivery mechanisms and acquisition of hardware and software infrastructure, 6) managing the customer relationship.

Chapter 5 describes the research methodology applied in the empirical part of the study. In order to increase the reliability and validity of the research, the inductive approach is conducted and its suitability for the examination of the research questions has been analysed. This study investigates the research questions by performing qualitative data collections and analysis techniques.

Chapter 6 presents analysis of the research data. The conclusion of the study, discussion and implications are also presented.

Chapter Two

2. Literature Review

A literature review was conducted in order to achieve the following objectives: identify the research gaps, determine both the research problem and questions, formulate the objectives of the study, acquire understanding of the research area under investigation, develop a theoretical background for this study, identify a suitable research methodology and finally develop a conceptual framework for this study.

2.1 History of open source software

The history of open source software dates back to the early 60's when software was regarded generally as scientific knowledge to be studied and shared freely among peers. Early vendors of large-scale commercial computers such as IBM, Burroughs and Honeywell,[12] used to ship their products with software which was free ((Carlo & Jesús, 2000), (DiBona & Ockman, 1999), (Evers, 2000)). It was normal for software to be freely shared among users and its code made public. However, this situation changed in the late 1960s after the ``unbundling" of IBM software. By the mid1970s, the drift was towards a more proprietary view of software. Commercial value was attributed to software, users were no longer free to redistribute it, source codes were no longer available and therefore users could not modify programs.

In late 1970s and early 1980s, the free software philosophy was reborn; Richard Stallman, formerly a programmer at the MIT Lab, resigned, and launched the GNU Project and the Free Software Foundation (FSF) - an organization dedicated to producing software that granted users four basic freedoms:

1. The freedom to run the program, for any purpose.
2. The freedom to study how the program works, and adapt it to the users' needs.
3. The freedom to redistribute copies of the software

[12] Early History of Computing Nathan Ensmenger, Ph.D.April 20, 2007 http://www.fi.edu/learn/case-files/hci.html

4. The freedom to improve the program, and release theses improvements to the public, so that the whole community benefits.

In order to satisfy the conditions listed above, Users of a piece of software must have access to its source code.

The GNU project resulted in the creation of a free operating system ((DiBona & Ockman, 1999), (Carlo & Jesús, 2000)). The GNU General Public License (GPL) was used as a legal tool to ensure that the software produced by GNU remains free, and also promote the continuous production and distribution of free software. Concurrently, the Computer Science Research Group (CSRG) of the University of California at Berkeley was improving the UNIX system .With funding from DARPA contracts, and contributions from a global network of UNIX programmers, the BSD Unix operating system was created in 1981((Weber, 2004), (Carlo & Jesús, 2000)). Initially, the software was not redistributed to non holders of a Unix AT&T licence. But in the late 1980s, it was finally distributed under the BSD licence which was an open source licence. Unfortunately, until 1988, use of BSD Unix required an AT&T license, since each release included AT&T source code. However, with increase in licence cost, coupled with the fact that some vendors only wanted the BSD originated components, it became necessary to rebuild some parts of the kernel and other utilities without using AT&T code. In June 1989, UCB published Networking Release 1 containing their TCP/IP networking system for the first time without any AT&T code, and released under an open license that allowed free source code modification and distribution. And later, following an initiative led by Keith Bostic[13], a completely new UNIX was developed that did not include any AT&T code and was released as Networking Release 2 under the same open license[14]. This generally formed the baseline for development of several other free software versions, such as FreeBSD, NetBSD, and OpenBSD (McKusick, 1999).

Developments in internet technology and the use of USENETS helped to build strong user communities and coordinate spatially distributed efforts. During the 1990s, much of the software already developed was integrated, consolidating the work of many isolated groups to produce more

[13] Keith Bostic was a member of the Computer Systems Research Group (CSRG) at the University of California, Berkeley and one of the principal architects of the Berkeley 4.4BSD and 4.4BSD-Lite releases. He also led the effort at CSRG to create a free software version of BSD UNIX, which helped allow the creation of FreeBSD, NetBSD and OpenBSD. http://www.bostic.com/keith.html

[14] http://www.livinginternet.com/i/iw_unix_war.htm

complex software systems and products. As a result of this integration, complete environments were being built on top of UNIX using open source software. The software industry witnessed a further change in the early 1990’s when in Finland; Linus Torvalds implemented the first versions of the Linux kernel. Later with the collaboration of many other programmers, more utilities were added to the kernel to complete the GNU/Linux operating system. The Linux kernel and the GNU applications implemented on top of it were covered by GPL and are being used today in many GNU/Linux distributions such as Slackware, Debian, Red Hat, Suse and Mandrake (Linus & David, 2001).

Within the past two decades open source projects have produced some of the most useful and finest quality software with some of them becoming the product of reference in their particular domain. Examples are Apache which is widely used as a WWW server, Perl, XFree86 which is the most widely used X11 implementation for PC-based machines, GNOME and KDE both of which provide a consistent set of libraries and applications to present the casual user with an easy to use and friendly desktop environment, Mozilla and OpenOffice, which is a high quality desktop office application. There is a clear evolution from infrastructure oriented software designed mainly for technical users to more mainstream user oriented software and enterprise related software. In addition to this, the motives behind open source projects has generally shifted from pure altruism and the quest for innovation to commercial and business oriented.

2.1.1 Free / open source software.

The terms; free and open source software are often used interchangeably. They are both similar in many ways - both focus on the free redistribution of programs and the requirement to make source code available - the licensees of both software are therefore entitled to access, use, copy, modify and distribute Free/OSS source and binary code without making royalty payments to the licensor, and to combine the software with other software code. (Murray & Duncheon, 2006), however point out some subtle differences between open source and free software. They contend that Open source software may be licensed with fewer restrictions on its downstream distribution. Open Source licenses may be “copyleft” licenses but, alternatively, may permit the licensee to make his/her modifications to the code private (“Academic” or “Non-viral” license). For this reason, and given its founders’ desire to support commercial adoption of Free/OSS, Open source software generally is seen as more commercial-leaning than Free Software.

According to (Kogut & Metiu, 2001), the Opens source movement emerged as a result of a split-off from the Free Software movement in 1997 and the term open source was coined in February 1998 by members of the Free/OSS Community who feared that the “copyleft” or “viral” characteristics of Free Software would restrict Free/OSS from being used by commercial enterprises – and, consequently, would limit the growth and adoption of Free/OSS. They embarked upon a marketing campaign to promote free software, which led to the development and use of a new term to describe the software they were talking about: “Open Source”. The main difference between the two types of software lies in their focus and the branding. While Free Software advocates believe that freedom *equates to* value, Open Source advocates believe that freedom *leads to* value. Both, however, agree that software freedom is very important. One of the reasons for the creation of the Open Source "brand" is that the term "Free Software" is ambiguous in English. Unlike some other languages, the word *free* in English represents two distinct concepts:

- Available at no cost
- Possessing freedom

While the second definition is the appropriate one in terms of Free Software, it is quite easy for it to be confused with, for the first. The term "freeware" is typically used to describe software distributed using the first definition. It can be downloaded and run without a fee being charged. Recently, the term *Software Libre* has become popular because the Spanish translation of "Free Software" is not as ambiguous, since Spanish distinguishes between "free as in freedom" (*libre*) and "free as in price" (*gratis*).

2.2 Motivation to contribute to OSS project

The Open Source dogma thrives on the idea that by enabling an environment where users who are interested in the use and production of software can collaborate and share their ideas, good quality software can be produce and put at the disposal of humanity while encouraging innovation and bringing satisfaction to participants in the open source community. In order to determine why the OSS development model has been so successful, it is important to understand the factors which motivate developers to participate in a project which is time and effort consuming only to give away the code and reveal proprietary information freely to strangers.

(Karim & Robert, 2003), investigate the intrinsic (the performance of an activity for its inherent satisfaction rather than for some separate and external consequence such as force, reward or pressure) factors which motivate participants to contribute to open source projects. (Lindenberg, 2001), classifies intrinsic motivation into two categories: enjoyment based and obligation/ community based intrinsic motivation. Enjoyable activities are thought to provide feelings of creative discovery when a challenge is overcome and a discovery resolved ((Karim & Robert, 2003), (Csikszentmihalyi, 1990)). The manner in which developers of free/OSS engage in free/ OSS projects strongly correlates with this theory. Programmers voluntarily choose to participate in a project and in a given capacity which both challenges and stimulates them (Karim & Robert, 2003).

(Lindenberg, 2001), also points out that, individuals may be motivated to act in a particular way because of a need to be consistent with the norms of a group or community. The obligation/community based motivation factor is strongest when personal gain seeking by individuals within the said community is minimized. (Karim & Robert, 2003), observed that a significant proportion of participants in free/OSS project felt a sense of obligation to give something back to the free/OSS community in return for the software tools it provides. In free/OSS communities there is also a strong sense of community identification and adherence to norms of behaviour - the “hacker” identity connotes a strong fondness for solving coding problems while having fun and sharing code at the same time - private-gain seeking within the community is minimized by adherence to software licenses like the GPL and its derivatives (Karim & Robert, 2003).

However, the factors which motivate individuals to participate in Open source projects are not always simply altruistic or intrinsic. Studies on the labour economics in community Open Source development reveal that a surprisingly high amount of voluntary work goes into the development of open source. (Lerner & Tirole, 2000), surveyed thousands of excellent programmers who contribute freely to open source projects and concluded that altruism is not the only reason for their devotion. Developers contribute to open source projects not only for the personal gratification that comes from increasing their reputation among peers ((Scacchi, 2004), (Raymond, 2001) and (Pavlicek, 2000)) but also use it as an opportunity to document their technical capabilities and improve job prospects with future employers.

(Lerner & Tirole, 2002), considered a theory of cost and benefit in explaining why programmers choose to participate in free/OSS projects. According to them, as long as the benefits exceed the costs, the developer will contribute. They propose that the net benefit of participation consists of immediate and delayed payoffs. Immediate payoffs for free/OSS participation include being paid to participate and the user’s need for particular software ((Raymond, 2001), (Franke & Hippel, 2003)). (Karim & Robert, 2003), explains that firms may hire programmers to participate in free/OSS projects because they are either heavy users of free/OSS-based information technology infrastructure or providers of free/OSS-based IT solutions. While delayed payoffs may take the form of skill advertisement through participation in OSS projects by contributing codes and also by using the community as a means to gain practical experience and improve their programming skills (human capital). The interactive process of peer review improves both the quality of the code submission and the overall programming skills of the participants (Krogh, Spaeth & Lakhani, 2003).

While many authors have studied individuals' motivation to contribute to open source software development, (Iansiti & Richards, 2006), examined the motivations of large companies and IT (Information Technology) vendors, to invest in OSS. They studied why companies who historically have been selling proprietary software would invest in OSS, also where they chose to invest and the characteristics of the projects to which they contribute. They found out that IT vendors were generally more interested, and invested in money driven (high impact OSS projects that can serve in a complementary fashion to draw revenues to their own (largely proprietary) core businesses projects, while very little investment was made on community driven OSS projects. Novel business models are being developed where companies develop software in tandem with a community, give it away for free and then offer fee based professional services including training, support and consulting for that software. Other companies encourage their employees to participate in open source projects so that they can influence those projects to suit their needs or yet in order cases such as with IBM, it may be used as a competitive strategy. By investing in open source software, they commoditise specific software products of rival companies.

2.3 Evolution of open source software

Much research have been conducted in a bid to understand the factors which drive open source communities to succeed in producing high quality and complex products in a rather extemporized development environment. In his book: *The cathedral and the bazaar*, (Raymond, 2001), used a simile to describe the chaotic production process of OSS (bazaar style), and contrast it with the carefully planned and tightly controlled process of producing proprietary software (cathedral).

Although this analogy might be a bit extreme, it portrays a major difference between the two types of software creation models: strong powerful management on one side and loosely related developers and users organized in several thousand seemingly independent projects on the other hand. According to (Raymond, 2001), quality in software development projects is not guaranteed by rigid standards or autocracy, but by the rapid release and feedback from hundreds of users. He also concurs with Linus Torvalds conviction that delegation of project tasks such as debugging, development and exploration of design space, is vital in producing good quality software within very short time periods ((Kogut & Metiu, 2001), (Raymond, 2001), (Weber, 2004)).

However, (Bezroukov, 2002), considers the presented bazaar model as a "too simplistic" view of the open source software development process. Instead he tries to explore links between open source software development and academic research as a better paradigm. According to him, it should be viewed as a special case of academic research.

(Kumiyo,Yasuhiro, Yoshiyki, Kouichi & Yunwen, 2002), studied the OSS development process of four OSS projects and contends that, the community is not as chaotic as it may seem at first glance. They examined the evolution of OSS systems and their associated communities and deduced that while collaborative development within a community is the essential characteristic of OSS projects, different collaboration models exist, and that the difference in collaboration model results in different evolution patterns of OSS systems and communities. They classified OSS communities into three categories based on their goals: *exploration oriented*, *utility oriented* and *service oriented*.

- Exploration Oriented: This category is typified by GNU software where the prime aim is to encourage the improvement of software development collectively through the sharing of innovations embedded in freely shared OSS systems. It is a classical illustration of the culture of scientific research community as pointed out by (Bezroukov, 2002; Bezroukov, 1990). Due to its free access, it can stimulate other developers to think of innovative ideas that otherwise would not have been born, and it enables others to go further by stepping on the shoulders of the previous developer through reusing the open source code. The quality Requirements for such projects are often very high. Once the system is released publicly, it becomes a learning resource for thousands of software developers. Therefore, this type of software is developed and maintained by expert programmers ("Project Leaders"), who often are the original developers and keep a tight control over the system in order to maintain the integrity of the system so that it reflects its original design goal. Contributions made by the community at large exist as feedback and are incorporated only if they are consistent with the ideas of the Project Leader (Fielding, 1999). The success of such OSS projects depends greatly on the vision and leadership of the Project Leader. However, if the vision of the Project Leader conflicts with the needs of the majority of the OSS community members, the project may diverge into sub projects. A new OSS project and community will be spun off the original one and embark on a similar but different project.

- Utility-Oriented OSS: the Linux operating system (excluding the Linux kernel, which is Exploration-Oriented) provides a good example of a utility oriented OSS; its main objective is to fill a void in functionality. Most of such OSS systems consist of many relatively independent OSS programs, such as the device drivers in the Linux operation system. These programs are developed because the original developers cannot find an existing program that fulfils their needs completely (Raymond, 2001). Instead of waiting for a proprietary developer to provide these functionalities, competent software developers initiate a project to develop these needed functionalities. In this kind of projects, timeline is very important since the project is often motivated by an emergent and practical need. Moreover, because the development is driven by an individual need, developers are concerned with developing an operational system rather than delivering a refined solution as in the exploration-Oriented type. This type of project is characterised by a proliferation of divergent sub projects all resulting in programs with similar utility. This type of OSS software development is a typical bazaar style project with very limited centralized control. In this type of projects, developers are usually only interested in programs which satisfy their particular needs. They therefore remain peripheral to the system while passive users, because of the proliferation of multiple alternatives and different versions to a piece of functionality, may need the help of specific distribution packages assembled for them by more technical members, to be able to identify and use the software they need.

- Service-Oriented OSS: Examples of this type of OSS projects are PostgreSQL system and the Apache Server. They aim at providing stable and robust services to both the developers and users of the OSS systems. In a Service-Oriented OSS system, because the population of stakeholders is much larger than that of the OSS community, any changes made to the system have to be carefully considered so the services which it provides to its users are not disrupted. For this reason, Service-Oriented OSS is usually very conservative against evolutionary and rapid changes and because of its conservative nature; the control style of Service-Oriented OSS is neither Cathedral-like (which is too restrictive as evolution depends on the ideas of the project leader) nor Bazaar-like (which is not very stable as it permits too many rapid changes). On the contrary, Service-Oriented OSS is often collectively controlled by a group of Core Members. There is no single Project Leader, and changes are sanctioned only after a group (council) of core members approve it. Furthermore, the membership of the Council is not fixed. Most OSS communities of this type have a mechanism of accepting new council members whose contributions and competences are well recognized and who are trusted by community members.

The nature of an OSS project is however not necessarily fixed over time. (Kumiyo, Yasuhiro, Yoshiyki, Kouichi & Yunwen, 2002), point out that while most OSS project are initiated as exploration oriented, over time, as the projects reaches a more mature and stable phase, it generally progresses towards a service oriented project

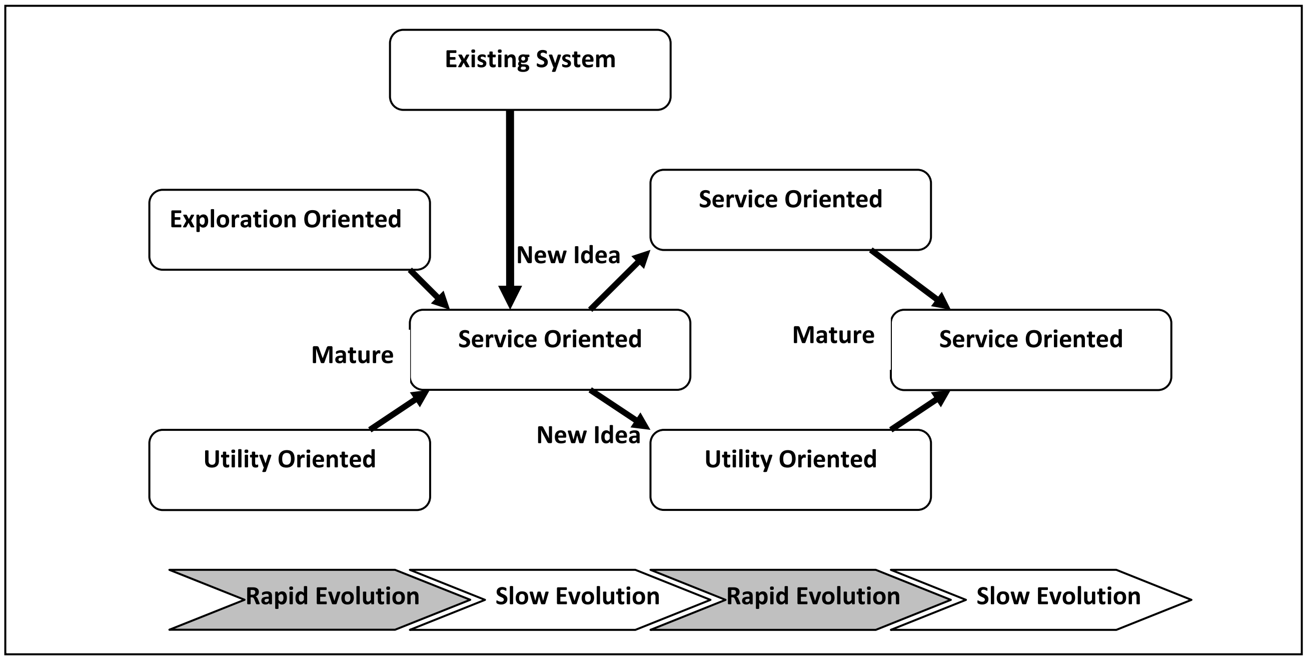


Figure 2.1 Evolution pattern of OSS projects

Source: (Kumiyo;Yasuhiro;Yoshiyki;Kouichi;& Yunwen, 2002)

The focus of the particular project determines the evolutionary pattern of the OSS system and community. It also determines the constitution and role of the community members. While all open source communities do not have the same roles, all have a project leader who is usually the person who initiated the project. Nevertheless, most often, the leader neither has a grand plan for the project at the beginning nor does he dictate the evolution of the system. One main contrast between OSS projects and the proprietary software development process is that members of the OSS project assume certain roles by themselves according to their personal interest, rather than being assigned a task by someone else. (Kumiyo, Yasuhiro, Yoshiyki, Kouichi & Yunwen, 2002), identified the following roles which members of an OSS community may occupy:

Passive User: They just use the system in the same way as most users use commercial software; their main attraction to the software is its potential high quality, low cost and malleability.

Reader: Readers are active users of the system; they not only use the system, but also try to understand how the system works by reading the source code. Readers are like peer reviewers in traditional software development organizations.

Bug Reporter: They identify and report bugs; they do not fix the bugs themselves, and they may not read the source code either. They assume the same role as testers of the traditional software development model.

Bug Fixer: They fix bugs which are either discovered by themselves or reported by Bug Reporters. They have to read and understand a small portion of the system's source code where the bug occurs.

Peripheral Developer: These members occasionally contribute new functionality or features to the existing system. Their contribution is irregular, and the period of involvement is short and sporadic.

Active Developer: Active Developers regularly contribute new features and fix bugs; they are one of the major development forces of OSS systems.

Core Member: They are responsible for guiding and coordinating the development of the OSS project. Usually they are part of the group that initiated the project and have made significant contributions to the development and evolution of the system.

These roles may not be present in all communities. While in some communities some of these roles may be merged together, activities performed by the members is generally similar in most communities.

Although a strict hierarchical structure does not exist in OSS communities, the structure of OSS communities is not completely flat. Depending on the role they play, members can exert different levels of influence in the community. Project leaders have the most influence while passive users have the least. Nonetheless, they still play an important role in the whole community because, although they do not directly contribute to the development of the system technically, their very existence contributes socially and psychologically by attracting and motivating other more active members. It is also important for the community to maintain a balance in its composition of all the different roles otherwise the community will not be sustainable. The roles are neither pre-assigned nor permanent, members likelihood of advancing through the community hierarchy depends on their level of commitment in the project. These role migrations bring about the evolution of the OSS community (Brian, 2006). As community members change the roles they play in the community, they also change the social dynamics, and reshape the structure, of the community.

(Kumiyo, Yasuhiro, Yoshiyki, Kouichi & Yunwen, 2002) state that the evolution of an OSS community is determined by two factors: the existence of motivated members who aspire to play roles with larger influence and the social mechanism of the community that encourages and enables such individual role changes. For an OSS system to have a sustainable development, the system and the community must co-evolve. GIMP (Gnu Image Manipulation Program at http://www.ghnp.org/) is an example of a project which stalled for a while because the community lacked qualified members who could continue the work initiated by the community leaders. For the long term sustainability of an OSS project, Project Leaders and Core Members of an existing OSS community should not only focus on the evolution of the OSS system itself, but also strive to create an environment and culture that fosters the sense of belonging to the community and mechanisms that encourage new members to move toward the centre of the OSS community through continual contribution.

(Brian, 2006), examined how it can be possible to seamlessly coordinate, produce and test software in such heterogeneous and distributed environment. He points out that the most successful Free/OSS products – the Linux operating system, the Apache web server, the Mozilla browser, the GNU C compiler, the Perl scripting language, and MySQL database management system – are all examples of horizontal infrastructure software. Their requirements are part of the general taken-for-granted wisdom of the software development community. (Kogut & Metiu, 2001), asserts that, the modular nature of the kinds of software which OSS communities specialise in is a major factor in their being able to release high quality software products.

2.4 Open source adoption and diffusion

A previous research carried out by (Calpy & Chenogorov, 2009), investigated some of the major factors which deter organisations from adopting open source software in spite of all its obvious advantages. Some of the major factors which were revealed include:

- The uncertainties about adequate support.
- The lack of reliable references for software in particular domains.
- Issues related to Intellectual Property Rights (IPRs).
- To a much lesser extent, security fears, fears of incomparability with existing system and fault tolerance with OSS.

Among the different categories of software studied, the most popular were Data Storage and management software, Internet software and Middleware software. In the first category, Lucene and MySQL were most popular while Joomla was considered a very good dynamic portal engine and content management system.

Open source adoption has been particularly dominating in internet and server side software projects[15] such as Apache. Open source is also making significant inroads in enterprise software and many establish software vendors such as Sun microsystem and IBM are developing new strategies to tap into the OSS potentials ((Sarkinen, 2007), (Evers, 2000), (Koski & Kretschmer, 2007), (Iansiti & Richards, 2006), (Hohensohn & Hang, 2003), (Krishnamurthy, 2003)) . Europe has being at the forefront of OSS adoption. A 2008 forester research showed that France is amongst the leading adopters of OSS[16] while Asia, Latin American countries such as Brazil, India and Argentina and Australia are developing policies to facilitate Open source adoption and diffusion.

[15] http://news.netcraft.com/archives/web_server_survey.html
[16] http://www.forrester.com/rb/Research/open_source_adoption_notes_from_field/q/id/46279/t/2

2.5 Open source business models

For the purpose of this study, a business model is defined as architecture for product, service and information flows, including a description of the various business actors and their roles; and a description of the potential benefits for the various business actors; and a description of the sources of revenue (Timmers, 1998)

In the preface of Setting up shop (Hecker, 1999), Franck Hecker points out the challenges which software companies face in today's business environment as they attempt to expand their business influence in the industry, manage the competition and improve revenue returns even as they seek to keep their employees motivated. Software businesses have to deal with issues such as growing their product lines, monitoring and enhancing product quality, supporting old and new releases while staying on the edge of innovation. This situation is even made more compounding today with the prevailing economic atmosphere which has pushed most enterprises to rethink their business strategies and fine-tune their IT budgets. (Hecker, 1999), proposes open source software as a solution to most of the above challenges. He claims that;

> These software business challenges are interconnected in two ways. First, most if not all are functions of constrained resources. Few companies have enough people, money, or time to do everything that needs doing, especially when competing against larger companies with greater resources. Second, a strategy exists to address all these challenges at once: turning some (or in exceptional cases all) of a company's software products into open-source ones.

While it is quite obvious that based on the quality of OSS, its compatibility with other software systems and applications and above all the low cost of acquisition, (Hohensohn & Hang, 2003), offers an opportunity for adopter companies and institutions to stream line their IT costs-centre budgets, It also presents an opportunity for software development and vendor companies to reduce time to market cycles and the total cost of production and therefore improve profitability (Riehle, 2007).

However, choosing a sustainable open source business model is more complex than just choosing the software licence. An open source business model comprises of different components depending on the software and the needs of the consumers. Therefore, the vendor needs to understand the implications of each component and how they can be applied to open source software in question

(Madanmohan & Pal, 2002), proposed a framework consisting of four ways that a commercial organisation can compose its open source practice. They suggest that based on its particular characteristics - age, experience, resources- the nature of the market in which they are operating and the type of software products or services which the organisation wishes to offer they may opt for either:

- Single OSS Initiative - Single Market: This is a strategy in which the company develops expertise and focuses in one market and uses only one open source technology as the platform for building products and solutions. Companies employing this strategy are usually Single-Product Companies, or those having a portfolio of related products for a single vertical market. Most of the open source initiatives start in this manner as it is well suited for small companies which do not have a large workforce. Upon maturity and stability of the software, they start exploring other markets/domains where the software can be used.

- Single OSS Initiative – Multiple Markets: The companies using this strategy leverage investments made in the open source initiative across multiple products for different vertical market segments or domains. The selected open source initiative results in a platform which can be used to build various commercial products for different market needs. This Model seems to be a natural progression from the first case. The strategy is suitable for a company that is operating in multiple markets with specific offerings in those markets and should have significant engineering resources to support an open source initiative and collaborate with the public-domain community

- Multiple OSS Initiative – Single Market: This model does not appear to be very profit oriented ad the company invests considerable amount of resources to initiate and manage multiple open source communities only to leverage it for its products/services for only one domain/vertical markets. (Madanmohan & Pal, 2002), stress that companies that are likely to adopt this model are those, who are world leaders in a particular product/technology, and dominate the markets. An example could be Nokia, which uses multiple open source initiatives to add value to its products and solutions for the Telecom market.

- Multiple OSS Initiative – Multiple Markets: The companies following this strategy are typically large companies, who can sustain multiple open source initiatives, and leverage them across multiple markets.

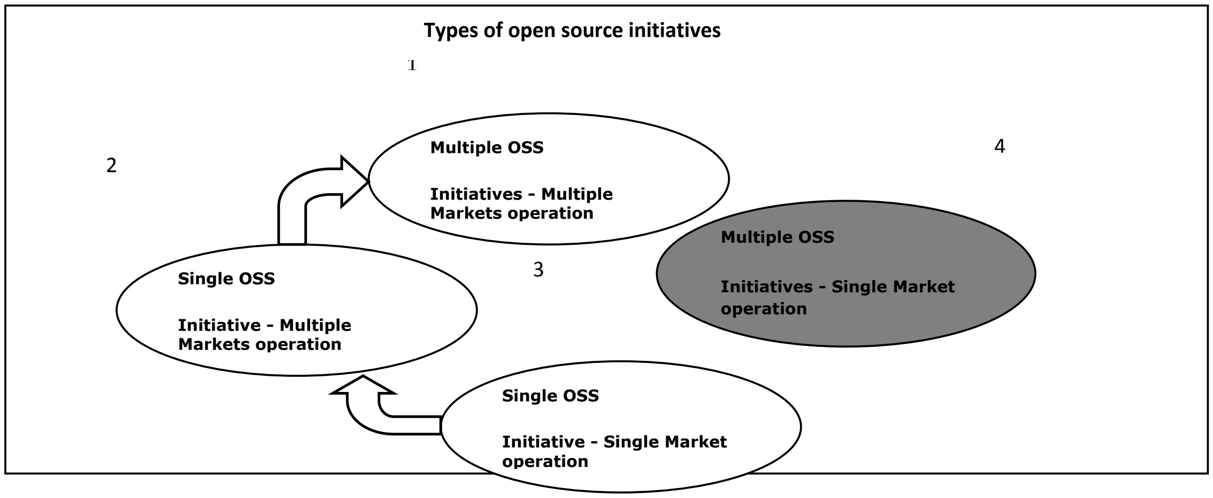

Types of open source initiatives
1
2
Multiple OSS Initiatives - Multiple Markets operation
4
Multiple OSS Initiatives - Single Market operation
3
Single OSS Initiative - Multiple Markets operation
Single OSS Initiative - Single Market operation

Figure 2.2 Evolution of commercial open source initiative

Source: adapted from (Madanmohan & Pal, 2002)

In other words, to ensure success it is important for an organisation to choose the appropriate strategy as this will define the rest of the open source strategy for the firm. However, whatever model a firm adopts at any given time is only a reflection of its circumstance and the opportunities and challenges posed by the business environment at that time. As these parameters change, the chosen strategy evolves with it. (Madanmohan & Pal, 2002), considered the fourth scenario (from the diagram above) to be a special case most suited for some companies as mentioned above.

The idea of a company making profit from software by giving the code out for free may seem paradoxical especially considering that traditional proprietary software vendors have realised most of their business success by protecting their source codes and capitalising on its licence sales as a prime source for generating revenue. However, many researchers and companies have found ways to navigate around this “absurdity” to develop viable business models based on open source software.

(Chang, Mills & Newhouse, 2007) elaborated on five business models for open source entrepreneurs. While their list is by no means exhaustive, they claim that these models have been the most successful and sustainable way of generating revenue using the open source software.

1. **Support and Service Contracts:** Depending on their experience and amount of resources, most open source businesses adopt more than one business model in order to ensure their sustainability and profitability in the dynamic set up of the software business world. As such, it is common to find the same company listed in different studies under two different business models (Chang, Mills & Newhouse, 2007), (Daffara, 2009)). While different variations of implementing the support and service contract model exists, it generally entails a vendor company which offers support services for a software product or service which it may offer for a fee or that a customer may download from the internet for free.

    - **Examples:** Companies which adopt this model include, Red Hat, OpenLogic and GBdirect. Red Hat adopts a support-based subscription model for its OSS business. Customers pay for Red Hat Enterprise Linux, which is a tested, certified and stable version of its free and community-based Fedora Linux. The advantage this gives is that it guarantees a high level of deployment, scalability and security. Apart from this, support subscription allows users to download and install security patches, and provides 24/7 online and phone customer support. The customer can also get additional support services such as technical account management, development support, premium developer packages, discounted commercial software (JBoss), as well as bug fixes and troubleshooting for users' local nodes by opting for a premium support service contract which is more expensive that the basic support contract. (Daffara, 2009) prefers to classify companies such as Red Hat and Nerus as platform providers because unlike other support provider companies such as OpenLogic and GBdirect which thrive almost exclusively from providing a one-stop support on one or several separate OSS products (usually by directly employing developers or forwarding support requests to second-stage product specialists), these companies provide selection, support, integration and services on a set of projects, collectively forming a tested and verified platform. In addition to the provision of support for opens source software, most companies which adopt this model also offer services such as:

i. Consulting - helping the customer to understand the benefits and risks of the specific product.
ii. Integration work - integrating the open source solution into an existing environment, deployment, implementation and management of the solution.
iii. Training - providing workshops and/or on-site training to help a customer get up to speed on the open source product in question.

Because of the nature of open source products, most customers expect the vendor of services to be engaged with the underlying project and to be visible as having relevant knowledge of the software. A vendor can emphasize his expertise by active participation in the open source project and by being vocal through blogs and articles in relevant publications.

- **Revenue logic:** According to (Sainio & Marjakoski, 2009), revenue logic has its foundation on pricing which is the marketing instrument that creates money. In the case of Open Source software, the item to be monetized is generally different from conventional (proprietary) software .There is a shift from product pricing to service fee. The three companies mentioned above obtain revenues from:

Table 2.1 Revenue logic of Red Hat, GBdirect and OpenLogic

Red Hat	**OpenLogic**	**GBdirect**
-Subscriptions from Red Hat Enterprise Linux (RHEL) per system or server basis; -Subscriptions from commercial open source applications per system or server basis;	- Support contracts for commercial grade technical open source support services which are classified as: **Silver Support:** assistance with "how to" questions and bug resolution as well as advice and recommendations on selecting, installing, configuring, and integrating open source software.	- Support contracts for helpdesk advice and diagnostic research. - Support contracts for remote onsite intervention services

-System/Architecture management services; Red Hat enterprise virtualization services - Support services; -Red Hat Certification and Training services	**Gold Support:** In addition to problem resolution, get more detailed assistance with configuration reviews, performance tuning, architecture reviews, production planning, migration planning, and project selection. Includes up to 20 hours of consultative support per year. **Platinum Support:** remote monitoring of critical server infrastructure combined with proactive response by OpenLogic's open source support team to alerts and problems. This option comes with a quarterly health check of the customers' production infrastructure, including recommendations for tuning the system, improving performance, and planning for future expansion and up to 20 hours of consultative support per year - Consultation and training services	

While GBdirect and to some extent OpenLogic's pricing strategy are based on the amount of time spent by their staff to address a customer's problem which is covered by a support contract, Red Hat's pricing strategy is mostly on a subscription basis.

- **Advantages of the Support contract Business Model:** vendor has the advantage that the cost of support can be partially shared across customers and the open source community while the value proposition for the adopter is that they have a single point of control and cost for a large number of projects and thus reduce negotiation efforts which would have been spent on a large number of individual vendors.
- **Disadvantage:** The major disadvantage is the limited market.

2. **Split (Dual) Licensing:** this model exploits the possibility of distributing the same software code under a copy left e.g. GPL and commercial licence.
   - **Examples:** software vendors that adopt this model Include: Digium, for Astersik - an open source telecommunications software suite-, Trolltech, for Qt - a cross platform toolkit used to develop GUIs-, Sleepycat software, for Berkeley DB - a database system- and MySQL for the MySQL database system. MySQL database comprises a free, community edition and a commercial, certified "server edition". The MySQL Community Edition is available under the open source GPL license with both stable and beta software releases and users are free to use MySQL under the GPL while in the case where the user is developing and distributing open source applications under an OSI approved licence which is *not* the GPL, they may take advantage of the FLOSS exception of the GPL licence that allows specific licences to be used.
   - **Revenue logic:** MySQL obtains the bulk of its revenues from selling customers a commercial license which permits them to use the product without being constrained by the obligations of the General Public Licence. These customers are therefore free to include MySQL in their own products for resale or use without the fear of having to make the source codes of their derived works open. Since MySQL has full ownership of the MySQL code it is able to tailor its commercial licensing terms to meet the unique requirements of users interested in embedding or bundling MySQL. In addition to the sale of commercial licence, MySQL obtains significant profits from offering premium customer support.

Advantages of Dual Licence model:

- The vendor has the economic advantage of being able to disseminate the product at reduced costs, create an external ecosystem which could serve as a source of add-ons, increase visibility and enable self-segmentation of the market ((Daffara, 2009), (Chang;Mills;& Newhouse, 2007)).
- Dual licensing can also be useful in circumventing some of the incompatibilities between OSI-certified licences. (Elena, 2009), highlights this point by pointing at the complicated Mozilla licence policy.

- It allows clients to use and customise the software for further sales without licensing restrictions.

Disadvantages of Dual Licence Model:

- (Elena, 2009), points out that dual licensing may have a negative effect on community contributions to open source projects. She claims that by allowing some people to keep modifications private whilst others are forced to make their changes public, the community built around the software code is likely to consist of many more users than developers. Although for a specific type of business model (for example where market penetration is a significant goal) dual licensing can be an important part of a company's marketing armoury.
- (Chang, Mills & Newhouse, 2007), claim that some clients may get confused with the boundary between commercial and GPL licence under the same product, particularly if they switch from using commercial support to OSS support
- Another problematic issue with the Dual licensing model is that without proper guidance and review any product or organisation in the entire sales chain may be subject to licence and legal requirements.

3. **Macro R&D Infrastructure:** This model may be initiated as a project funded by an institution or organization in need of a new or improved version of a software package. The funding company may decide to sponsor a consultant or software manufacturer to do the work and later release the resulting software as open source in order to take advantage of the large pool of skilled developers who can debug and improve it.

- **Examples:** The Maemo platform, used by Nokia and OMII-UK are good examples of this model. According to (Daffara, 2009), Maemo has only 7.5% of proprietary source code leading to a reduction in costs estimated at 228 Million dollars and a reduction in time-to-market of one year. Another example is the Eclipse ecosystem, an integrated development environment originally released as Free Software by IBM and later managed by the Eclipse Foundation. Many companies adopted Eclipse as a basis for their own product, and this way, reduced the overall cost of creating a software product that provides in some way developer-oriented functionalities. OMII-UK is funded by EPSRC through the UK e-Science Core programme and it aims to be a leading provider of reliable interoperable and open-source

Grid middleware components services and tools to support advanced Grid enabled solutions in academia and industry. The OSS development is achieved by investing in community developers to produce the functionality required by their user community. (Chang;Mills;& Newhouse, 2007), classified OMII-UK as a Macro R&D model, because it: presents engineering challenges, by integrating software components in a single container and providing solutions to meet demands of such challenges. Furthermore, it offers a secure, robust and fully integrated Service Oriented Architecture for academia and industry in the UK and globally and also provides interoperable solutions.

- **Revenue logic:** Funding for carrying out such open source research project mostly come from funding bodies such as academic and research institutions and R&D departments of large organisations. The focus is usually not to directly market the products of the project, but to use it to drive innovation and cut cost.
- **Advantages of R&D infrastructure models:**
    - This model offers an opportunity for companies to cut costs while carrying out cutting edge innovation by using shared resources.
    - This kind of projects can easily attract funds from government, global partners or commercial organisations if they meet a specialised area where there are high demands for both R&D and investment.
    - It may promote collaboration and partnership, and organisations may merge together to form a powerhouse in a specialized area to attract more expertise and funding.

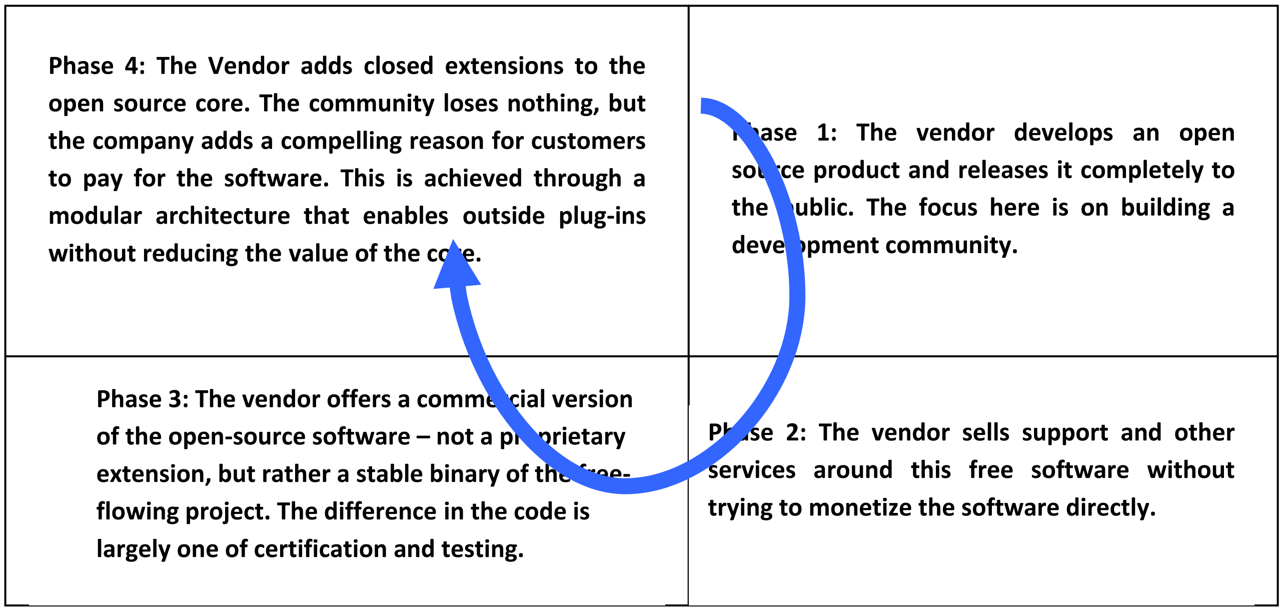
Disadvantages of R&D infrastructure models:

- The need to seek funding with regular intervals, and can create a sense of instability and insecurity.
- Might be difficult to integrate academic theories and industrial perspective in some organisations.

4. **Value-added close source:** This model has been described variously by different researchers and leaders in the open source industry (Freemium model): This refers to a business model which consists of withholding features from a free version of software and making them available only in a commercial version. (Lampitt, 2008) and (Daffara, 2009), have both referred to this model as the Open Core licence business model. Essentially it describes a model where a company uses open source software as a platform for developing a software

product or system and then releases this under two licences, one closed and another open it is very similar to the dual licence model (and is often confused with it). However, unlike the dual licence model, the discrimination between the two models is based on the fact that, the proprietary version of the open core model benefits from added features while for the dual licence model, the focus is on the "freedom" which the proprietary licensed software gives its users to keep their source code close. There is thus segmentation in the open core model based on features rather than on users.

- **Example**: vendors that use this model include Zimbra and XandrOS – XandrOS started off by operating a Split Licensing model similar to MySQL's however, from 2006 onwards XandrOS stopped releasing the open source version and now only distributes the commercial product, which contains proprietary software and some GPL software. In their commercial business model, XandrOS adopts "pay for software product" and "pay for services" and runs the two operations in parallel. Matt Asey[17] proposed an approach of developing and evolving an open source business model. He termed this the "phased Approach".



Phase 4: The Vendor adds closed extensions to the open source core. The community loses nothing, but the company adds a compelling reason for customers to pay for the software. This is achieved through a modular architecture that enables outside plug-ins without reducing the value of the core.

Phase 1: The vendor develops an open source product and releases it completely to the public. The focus here is on building a development community.

Phase 3: The vendor offers a commercial version of the open-source software – not a proprietary extension, but rather a stable binary of the free-flowing project. The difference in the code is largely one of certification and testing.

Phase 2: The vendor sells support and other services around this free software without trying to monetize the software directly.

Figure 2.3Matt Asey's four phased approach of developing open source business model

17 Matt Asay: A time to reap, a time to sow: A phased approach for open-source businesses

http://news.cnet.com/8301-13505_3-9945870-16.html

(Madanmohan & Pal, 2002), proposed a model in which a company may work with open source communities to build an open source platform and then build proprietary end user applications on top of this platform. They claim that historically, open source initiatives have had more success in the Infrastructure and Platform space, than with the end-user/GUI side. Most OSS projects have focused on software systems which can be used as backend solution to implement end user software products. They maintain that the primary audience of the open source software were originally software engineers and proposed the following steps for a business model based on open source initiative:

- Splitting the total product/solution into two parts - ‘platform’ and ‘end-user’.
- Create an open source community to develop the ‘platform’ part of the total solution.
- Develop the ‘end-user’ part of the solution in-house, while having close interaction with the open source community.

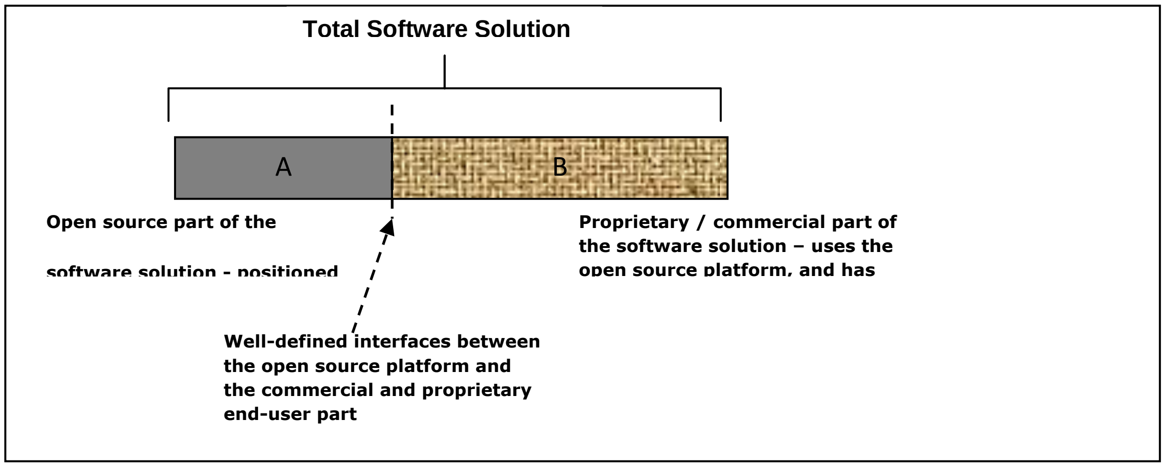

Total Software Solution
A
B
Open source part of the
software solution - positioned
Proprietary / commercial part of
the software solution – uses the
open source platform, and has
Well-defined interfaces between
the open source platform and
the commercial and proprietary
end-user part

Figure 2.4 Levaraging open source software to achieve software production efficiency

Source: (Madanmohan & Pal, 2002)

“A” represents the platform part of the offering which is open source, while “B” represents largely closed source end user applications. B can also include software services which a vendor may provide as part of customisation/porting of the open source platform software for a specific requirement and the dotted line represents the interface APIs that allow the non-open source software to interact with the open source software.

- **Advantages of Open Core Model:** The main advantages for the vendor using this model are: reduction of R&D and maintenance costs, visibility, increased dissemination, external ecosystem for add-ons, and self-segmentation of the market for the proprietary add-ons. While for the adopters, it is the freedom to opt for the Open Source edition if it is deemed sufficient.
- **Disadvantages of Open Core Model:** The major disadvantage is the difficulty for a product manager to estimate the right balance between open and closed parts while external groups may create substitutes for the proprietary parts thus eliminating its value to customers

5. **Community:** This model of OSS development and provision is not really a business model rather it is a method of assembling a community and obtaining resources from various stakeholder groups in order to produce and offer a general public good or service. This "business model" best fits to the original open source philosophy.
   - **Example:** The Apache Software foundation (ASF) is a prominent example of a community oriented OSS business model. Apache software foundation is a non-profit corporation created to support Apache software projects, such as the Apache HTTP Server. ASF produces and distributes software under the Apache licence which requires preservation of the copyright notice and disclaimer and unlike the GPL; it allows the use and distribution of the source code in both free/open source and proprietary software. Along with Red Hat/Fedora Linux, ASF is one of the largest OSS organisations, as evidenced by the 66.9 million sites using Apache as the web server[18].
   - **Disadvantage of community open source model:** The main weakness with the community oriented business model is that it relies on the community donation for its sustainability.

Research has identified several different ways by which companies make revenue using OSS. While most of them actually do generate income, they may not qualify as complete business models on their own. Most businesses combine some of these components to form viable offerings for which customers are willing to pay. ((Bonaccorsi;Rossi;& Giannangeli, 2004), (Karels, 2003), (Wichmann,

[18] Netcraft survey: http://news.netcraft.com/archives/web_server_survey.html

2002), (Daffara, 2009)) discussed comprehensively on this issues in their write ups. Some of these business components include:

- **Open Source Distributors**: These businesses generate revenue by directly monetizing the open source product itself. The open source licence does not forbid individuals or companies from demanding pay for the sales (distribution) of the software product. Businesses such as Linux distributors are probably the best-known firms in the product side. These firms aggregate, integrate and optimise the newest Linux files that are freely downloadable from the Internet. These activities add significant value and convert raw software fragments in a ready to install operating system, usually supplied on a CD Rom together with documentation. Niche and specialty Open Source distributors carry on the same activities on the code developed within a wide variety of Open Source projects. Walnut Creek CD-ROM was one of the earliest businesses to leverage free software, compiled public domain and freely available software for sale other examples of Open source distributors include Red Heart, Ubuntu and Slackware,. Distributions are sold by retailers who often combine their offering with other Open Source related materials, such as books or gadgets. This business model (accessorizing business model) has been successfully set up by O'Reilly & Associates, which publishes books that document and explain many Open Source program. The value in this model is the compilation of packages and easier access but now, this business model has been compromised by wide access to the Internet, especially with higher-speed access.
- **Selection/consulting companies:** These companies are not strictly developers, but provide consulting and selection/evaluation services on a wide range of project, in a way that is similar to the analyst role. They tend to have very limited impact on the Free Software communities, as the evaluation results and the evaluation process are usually a proprietary asset.
- **Training and documentation:** Typically, most open source projects do not produce standard documentation. Some companies such as OSS Nokalva offer to write documentation and manuals to any customer who may be interested in them along with on-line and physical training courses, integration, porting, consulting and testing services[19]. .

[19] OSS Nokalva, Inc. www.oss.com

These services are usually offered as part of a support contract, although recently several large scale training centre networks started offering Free Software-specific courses.

- **Legal certification and consulting:** These companies do not provide any specific code activity, but provide support in checking license compliance, sometimes also providing coverage and insurance for legal attacks; companies like OpenLogic employ tools such as the OSS Discovery that helps enterprises find the open source software embedded in their applications and installed on corporate workstations and servers. This enables the enterprise to better manage their open source usage and remain compliant with internal policies, regulations, and software license terms.

- **Product specialists:** Companies such as Compiere, Zenoss, Ultimate EMR, create or maintain a specific software project or very closely related software, and use a Free Software license to distribute it. Users freely acquire the software while the company generates revenue by providing support, training, consulting services custom development and maintenance. The company leverages the assumption, that the most knowledgeable experts on a software product are those that have developed it, and this way can provide services with a limited marketing effort, by leveraging the free redistribution of the code. The disadvantage of the model is that there is a limited barrier of entry for potential competitors, as the only investment that is needed is in the acquisition of specific skills and expertise on the software itself.
- **Platform providers:** Examples of companies which adopt this model include Red Hat Linux, Novell SUSE Linux, SourceLabs. These companies provide selection, support, integration and services on a set of projects, collectively forming a tested and verified platform. The main value proposition to customers comes in the form of guaranteed quality, stability and reliability, and the certainty of support for business critical applications and legal protection. While the major disadvantage is that platform engineering requires large R&D efforts even with shared resources.
- **Hardware integration:** A few vendors, such as VA Linux, began providing complete integration, with open source software pre-installed on hardware selected for compatibility and suitability to the operating system. Larger PC vendors eventually followed suit. IBM

later entered the market, offering Linux on its entire line of hardware. In 2002, it reported making $1.5 billion selling hardware running Linux[20].

(Chang, Mills & Newhouse, 2007), points out that many open source companies are in perpetual transition; as their surrounding circumstances constantly change, they also modify their business model in other to achieve a more sustainable mix. Examples include Xensource which was created in 2005 and operated mostly following the community model but currently, Xensource provides two licensing models, the first one through the GPL licence, which allows users to download, install, build from source and customise for personal or organisational uses. The second licensing model is through an Enterprise Linux (mainly Red Hat and SuSE) Licence, where clients can use this software if purchasing or subscribing to these Linux distributions. From April 2006, Xensource released their first commercial software package; Xen Enterprise which was based on development and improvement of Xen 3.0. Some other companies such as Sun microsystem and IBM have leveraged their respective dominance in a particular domain in the software industry and the special advantages which OSS offers, to operate multiple business model.

2.6 Open source software IPR – patents and copyright licence

2.6.1 Software patents

(Wilson, 2009), described a patent as a set of exclusionary rights granted by a state to a patent holder for a limited period of time. These rights are granted to patent applicants in exchange for their disclosure of the inventions, and upon payment of a yearly fee. Once a patent is granted in a given country, no person may make, use, sell or import/export the claimed invention in that country without the permission of the patent holder. But without a patent, anyone else can imitate their invention. Patent holders can also allow others to use their ideas for a fee.

In order to acquire a patent, an inventor must describe their invention and in doing so demonstrate that it is *new*, involves an *inventive step* and it *can be used as part of an industrial process* - meaning it is not just an intellectual or artistic endeavour.

[20] http://www.cioupdate.com/news/article.php/1574431

The patent right will lapse after a certain period set by the law, and can be allowed to lapse before that date, if the inventor wishes to stop paying the fees. Once a patent has lapsed, anyone can employ the ideas that are embodied by the invention.

The issue of software patent is very ambiguous in its interpretation and application in different parts of the world and even in the same regions over different time periods. In 1973, the European patent convention under article 53 identified certain types of inventions which must be excluded from patentability considerations. These include:

- discoveries, scientific theories and mathematical methods;
- aesthetic creations;
- schemes, rules and methods for performing mental acts, playing games or doing business, and programs for computers;
- presentations of information

By looking at the above exclusions, software programs did not qualify for patent in the European Union. However, with the creation of the European Patent Office (EPO) in 1978, and proliferation computers, coupled with the fact that across Europe, large research investments have been made into software development, the EPO started interpreting the 1973 exclusions more flexibly. The EPO responded to pressure from industry and started granting patents for what were essentially computer programs by allowing claims for processes that employed programmed computers[21]. This resulted in

[21] In 1978, Vicom (a software company) successfully appealed an application for a patent on a mathematical process embodied in a computer program that processed digitised images. The European Patent Office allowed the claim on the grounds that:

- Although mathematical methods cannot be patented, technical embodiments of them can be.
- Software that implements a technical embodiment of a mathematical method (or indeed any software) should be protected if it makes a technical contribution to the state of the art in the area of endeavour it inhabits. In 1998 the EPO revised its approach (in response to a patent appeal brought by IBM) and announced that in future software itself could be patented as long as it displayed a technical effect.

http://www.oss-watch.ac.uk/resources/softwarepatents.xml a proliferation in software patents in Europe to over 30,000 by 1997 and it was estimated at the time to increase by 3,000 patents annually[22].

In the United States, the position on the patentability of software has always been more permissive. Software was never excluded, and it was therefore never required to demonstrate a technical effect beyond the standard operation of a software program in order to qualify for consideration for patentability. All that was required for a program to be patentable was that it produced a useful, concrete and tangible result. This also explains the significantly higher number of software patents which exist the United States – about 15% of all patents (Bessen & Hunt, 2007).

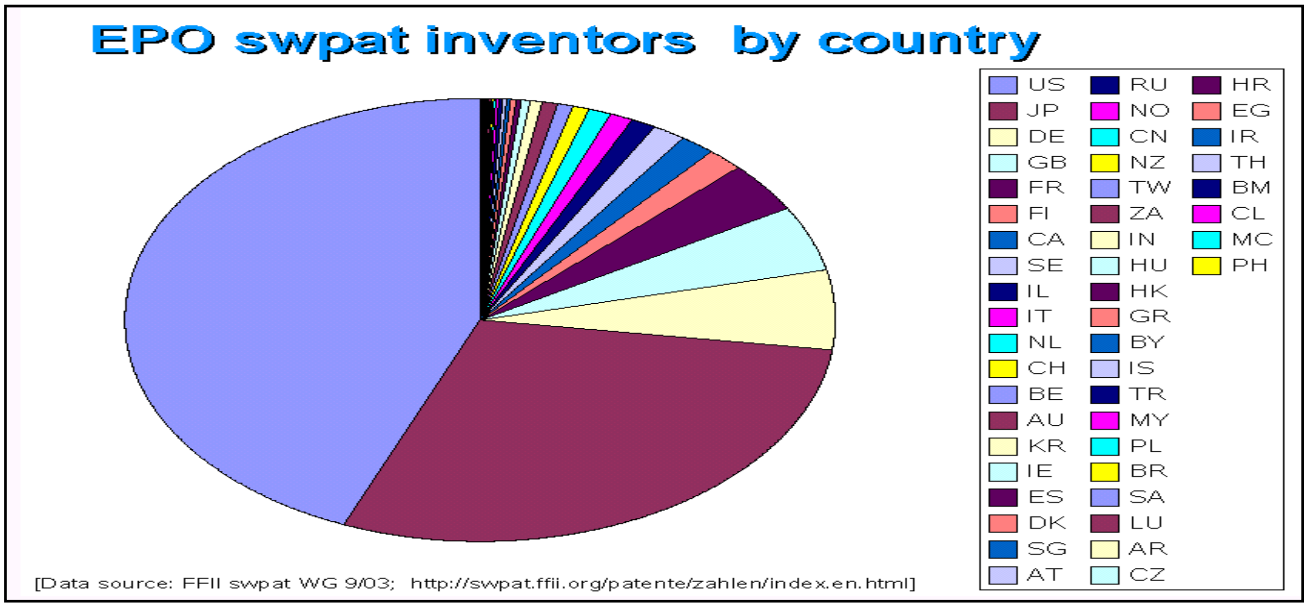


Figure 2.5 Chart of software patents by country

2.6.2 Open source licences

The issue of patent ownership and free/OSS licence is a very pertinent one. Licensing your code as open source may grant rights under patent law to the recipients of your code, and since the free and

[22] Software Patents in Europe: A Short Overview **http://eupat.ffii.org/log/intro/**

open source licensing model is designed to disseminate software and its associated licence as widely as possible, such a patent grant will also be, potentially, very wide both in the rights it grants and the audience to which it grants them. This viral property of OSS is meant to encourage improvement of the code, encourage public participation, and further innovation.

In this regards, OSS licences satisfy one of the major canons of patent law: that of moving information and technology into the public domain as rapidly as possible. However (Boettiger & Burk, 2004), point out that there is some danger that the open source approach might be seen to frustrate another goal of the patent system: that of providing an economic incentive or reward for innovation. Therefore OSS is on a fine balance between the two major tenets of patent law – that of furthering innovation and rewarding innovation.

(Seppä, 2006), notes that, the possibility of patenting and accidental patent infringement may lead to a strategic activity where the industry incumbents try to pool as many patents as possible in order to prevent new entrants into the industry. This kind of strategic patenting, he argues, is worthless from the point of view of society and welfare.

(Heller & Eisenberg, 1998), goes on to assert that the problem is particularly acute among enabling technologies which are themselves employed in the innovation process where the indiscriminate use of patents has burden research with high transaction cost and uncertainty because the tools of science have been privately appropriated and access to them is constrained by strategic intellectual property right ownership.

Increasingly, patent laws are attributing general principles and methods in software development to private ownership ((Bessen & Hunt, 2007), (Jullien & Zimmermann, 2007)). Large and Established enterprises see IPRs as incentives to compete in IPR portfolios. They accumulate patents to serve as strategic assets to protect them from competition, give them design freedom, offer complementary protection and form basis for new alliances ((O'Sullivan, 2002), (Välimäki, 2006)). (Rivette & Kline, 2000), puts it more succinctly, when he stated that:

> Whereas executives once feared that competitors might out produce or out market them, today they worry that rivals—especially in the booming e-commerce industries of the Internet—may secure the patent rights to the essential technologies or even to the fundamental business concepts that they need in order to be in business in the first place.

This fear of competition and the desire to monopolize industry wide ideas and their expression has pushed organizations into a race for larger patent portfolios which pose the greatest threat to software innovation ((Dixon, 2003), (Bessen & Hunt, 2007)).

2.6.3 IPR – Patents and copyrights

(Välimäki, 2001), proposes a framework for understanding the concepts of intellectual property right, copyright and patent. He notes that the difference between a patent and a copyright lies in the element which is being protected. While a patent protects an idea, a copyright protects the expression of the idea. Thus in this sense, a patent is a higher category over the Copyright although in practice, both IPRs often overlap as in the case of most software products. He notes that the choice between a copyright and a patent should be driven by strategy. While new patents signal continuous development to a firms competitors, they are also registered and hence easier to manage, trade, and used as a strategic asset. Copyrights on the other hand are more difficult to manage because while patents are clearly determined in the patent claims, there is no strict scope of copyright. However when considering simplicity, copyright law holds a clear advantage over patent. Copyright is granted automatically and for free if the work is considered original enough, whereas the patent process is expensive and tedious.

However (Jullien & Zimmermann, 2007), argue that software IPR protection is still not satisfactorily settled by the copyright protection because of the very specific nature of software products and the production conditions. While software products can be considered in the field of copyright because they are intellectual expression of ideas that are coded by the use of a specific programming language, with their specific vocabulary, syntax and structural rules, from a practical point of view, software programs can be considered a technology and should fall in the field of patents. The reason is that they aim at carrying out a given task relying on the resources of the computer in which it is implemented in or in the case of a software system, coordinate the running of the different components of the computer architecture.

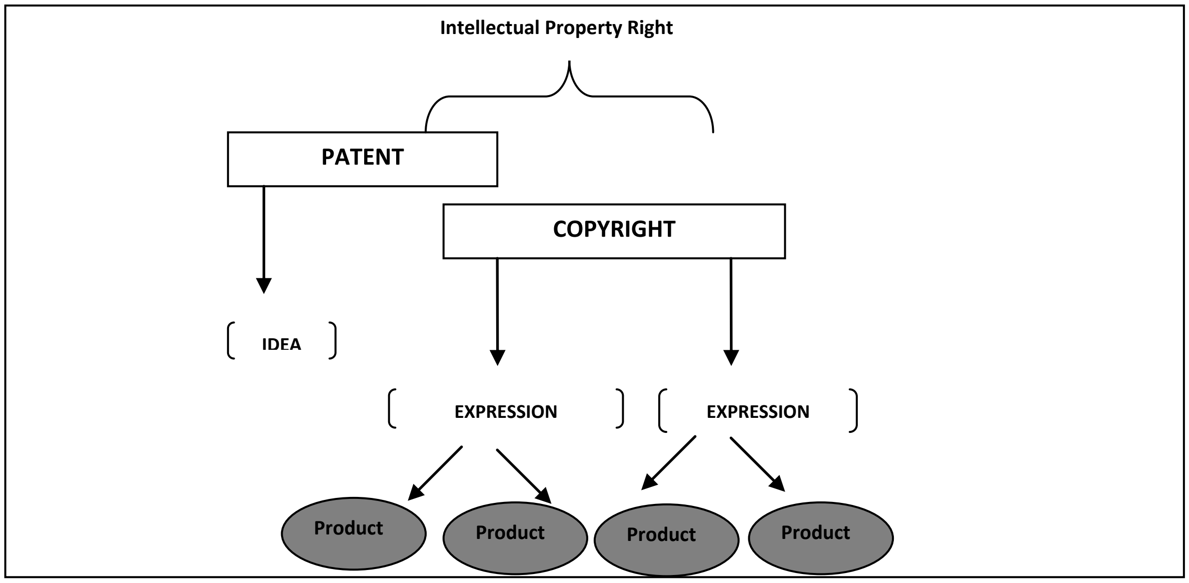


Figure 2.6 A disambiguation of intellectual property rights

Source: Adapted from (Välimäki, 2001)

2.6.4 Legal Issues with open source software license

(Boettiger & Burk, 2004) explains that proprietary practices have capitalised on the technical aspects of software which permits software to be written in a human readable "source code" and executed in machine readable "object code" format. The ability to separate and release software only as object code enables proprietary companies to capitalise on the software product, retain control over the way users use the software and discourage competition. The nature of software makes it very difficult for competitors to obtain the source code by reverse engineering of the object code and in most countries the proprietary companies can easily block any attempt at reverse engineering by imposing copyright licences which make such practice illegal (Jullien & Zimmermann, 2007). The problem with software protection through copyright is that copyright protects a given expression of the ideas and not the ideas themselves. Software producers are therefore not obliged to disclose the source code of the protected programs. This appears totally contradictory to the aims of intellectual property protection in so far as the owner of intellectual property is not constrained at all to reveal any information on the working principles of the protected program.

It is in the wake of this situation that the Free/OSS licence option was conceived. It aimed to preserve the diffusion of ideas and the combinatorial and cumulative nature of technical innovation, both in terms of concepts, tools and methods of coding ((Gomulkiewicz, 2002), (Jullien & Zimmermann, 2007)). In order to ensure the continues propagation of Free/OSS practice and prevent any private appropriation of part or all of Free/OSS code, the GNU-General Public License was developed. The GPL does not seek to force authors to renounce their rights to intellectual property it only compels them to relinquish the monopoly rent, which such rights would produce in a copyright regime. The main legal aspect is that, when a program is declared under GPL license, any code derived from it or integrating GPL code lines must also be available under GPL License. (Jullien & Zimmermann, 2006), assert that the GPL status is contagious in the sense that once it attaches to any number of lines of codes, it is automatically transmitted to the whole program into which they are incorporated. By opting for the GPL, authors authorize anyone who wants to make use of their work to do so under the sole condition that the new product must also circulate "freely". However the mere aggregation of another work not based on a GPL to a GPL licensed product on a volume of storage or distribution medium does not bring the other work under the scope of the GPL licence[23]. While GPL has an influential role in opening new principles of intellectual property management, it does not necessarily fit to the needs of many actors of this emerging open source world, particularly to those of the commercial firms that decide for different reasons to join the Open Source alternative. For this reason, many "hybrid" licenses have been designed in order to reconcile cooperative development and private interests in a variety of specific contexts. They involve different ways of combining the copyright and copyleft rules in different proportions ((Smets-Solanes & Faucon, 1999), (Muselli, 2002)).

(Gomulkiewicz, 2002), identified several shortcomings with the GPL licence. He claims that it is cumbersome and ambiguous and in many instances leaves the adopter wondering what is covered by the licence and what is not. It is also not clear whether some terms in the licence (such as the right to charge a fee for physically transferring a copy of the software) are advisory, a separate covenant or an additional condition.

Another important open source licence is the BSD Licence – an end user licence for the Berkeley Software Distribution of UNIX. Comparing the GPL to the BSD licence, (Gomulkiewicz, 2002), points out that in contrast to the GPL, the BSD is short and appears both in content and in form much like many mass market licences. Users of this licence also have the freedom to make

[23]Open Source Initiative OSI - The GPL: Licensing **http://opensource.org/licenses/gpl-2.0.php**

derivative works open or closed, to share or not share modifications and charge fees for the software or release it for free. However he points out that this licence also has some flaws. For example, it does not explicitly grant the right to modify software although this is implied.

(Malcom, 2002), points out that because open source licenses typically disclaim all damages, not just the standard disclaimers of consequential and special damages, it can be argued that an otherwise valid contract is rendered invalid since one party has no right to damages for breaches of the agreement by the other party. They claim that this lack of “mutuality” is something to consider when a Developer distributes open source software with its own software.

(Malcom, 2002), also point out that because most open source software is developed by multiple people and companies, it can be difficult to determine who the actual licensor is. The lack of a central licensor is very crucial because firstly, there may be no person or organization that can sue an End User or Developer who violates the terms of an open source license agreement. And conversely, there may be no one for an End User or Developer to sue if that becomes necessary or appropriate.

Open source software generally appears to be more vulnerable to litigation than Proprietary software because since the source code is available, it can be easier to determine whether or not copyright or patent infringement has occurred, whereas making only the object code available tends to hide this fact ((Boettiger & Burk, 2004), (Malcom, 2002))

The problem of choosing the right open source licence is currently very crucial as increasingly complex software products are built from the combination of elementary modules into a global architecture. (Gomulkiewicz, 2002), points out that this approach requires both an increasing recourse to a large scope of software components, portable and reusable in different contexts, and a growing proximity to the mathematical foundations of programming. This evolution makes the problem of the distinction between public and private property of modules and algorithms more acute.

(Rosen, 2004), asserts that what drives the licence selection process is the vendor’s business strategy. The open source model has two distinctive and related features: the use of collaborative development structures that extend beyond the boundaries of a single firm and the lack of reliance on intellectual property rights as a means of appropriating the value of the underlying technologies.

(Välimäki, 2005) categorizes Open Source license terms to those requiring:

1. Standard reciprocity Licences (meaning that distribution of the code is allowed only if the distributor keeps the distribution terms of the original code untouched, such as Mozilla public licence (MPL) and GNU LGPL). This licence type requires that source code with further development or changes cannot be closed. One of the most important features of such licenses is that if source code is combined with another source code then this licence does not apply to the new work anymore. This type of licence does not allow any kind of patents for the third party and is incompatible with patent licensing fees. (Malcom, 2002) claim that the Lesser General Public Licence is probably the most technically complicated open source license mainly because of its subject matter: the use of libraries by software programs. In its Preamble, the LGPL points out that if one takes software and links it to a library, the combined work is legally a derivative work of the library. Accordingly, using a library that is subject to the GPL would render the entire work subject to the GPL.

2. Strong reciprocity Licences, (meaning that in addition to the original code, even adaptations and derivative works are allowed only if the distributor keeps distribution terms untouched, examplse are: GNU GPL, Common Public Licence). This category of licence is stricter than the LGPL and it forms the basis from which the LGPL was derived. GPL is the most popular among OSS developers (SourceForge Web site has over 38,000 software packages available for download that are licensed under the GPL)[24] and therefore influences the rate at which a viable community is built around an OSS project.

3. Permissive licences (the terms of this licence allow free distribution, copying and modifications. Examples include: Apache, BSD, Artistic, Public domain, MIT, Zib, Python License and Academic Free License). This kind of licence allows the greatest degree of freedom as it permits the holder to offer software in both open and closed context and implement different mechanisms of charging fees. It is also less protective of the open source movement and more flexible toward End Users and Developers.

[24] www.sourceforge.net

The MIT license is very similar to the BSD license. The use of software subject to the MIT license requires a copyright notice, a disclaimer of warranties, and a limitation of liability. The software license is otherwise unrestricted.

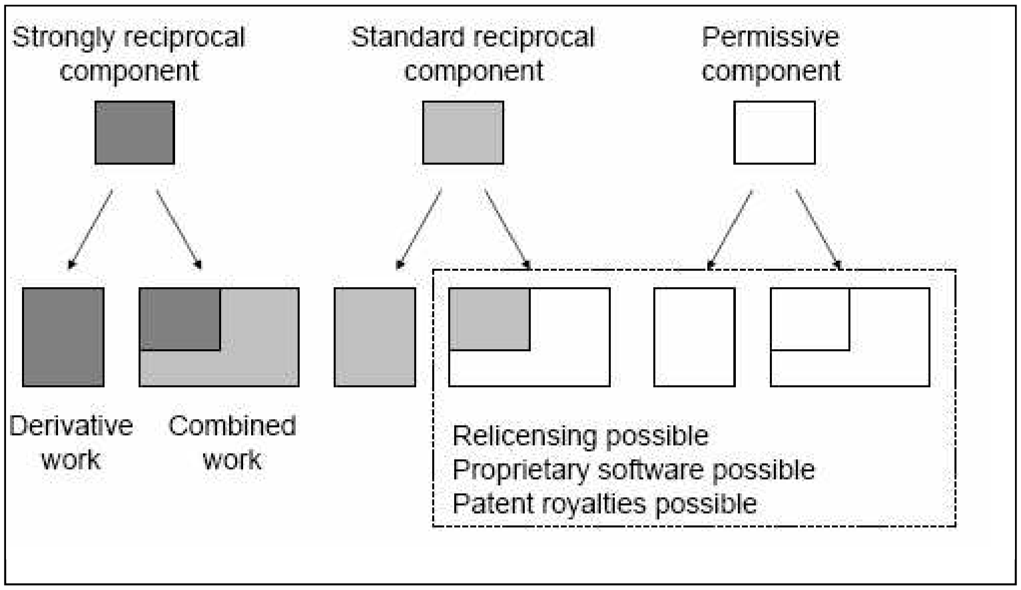

Strongly reciprocal component
Standard reciprocal component
Permissive component
Derivative work
Combined work
Relicensing possible
Proprietary software possible
Patent royalties possible

Figure 2.7 Functional differences of OSS licenses

Source: (Välimäki, 2005)

In most cases, the end users are not really impacted by the language of the LGPL, however, for developers who wish to use software which is licensed as open source, it is very important to read through the text of the particular licence carefully and put it in the context of the particular software and business model. For example Developers who use libraries subject to the LGPL need to consider the requirements of Section 6, most notably the requirement that the Developer must grant its licensees the right to modify the code for internal use and reverse engineer for debugging,[25] while Developers who want to use the open source software licensed under the GPL should carefully consider the risks associated with interoperability, as well as whether the GPL imposes restrictions on the Developer's proprietary code as a "collective work" of the open source software. They should specifically and technically examine how their proprietary software works with the open source software. Usually the risks are manageable while the benefits may be significant nonetheless it is always important for the Developer and adopter to be aware of the obligations which come with the licence.

[25] http://www.opensource.org/licenses/lgpl-3.0.html

2.7 OSS and cloud computing services

Some authors have claimed that Software as a Service (SaaS) along with proprietary software are the major competitors to OSS (Kepes, 2009). However, there are also many experts who believe that developments in SaaS technology and OSS need not be antagonistic. In fact, major successes have been realised by companies such as Zimbra, MindQuerry, and Arena Solutions (which produces and sells a hosted, subscription-based product lifecycle management (PLM) tool for manufacturing companies[26]). (Subramanian K. , 2008), argues that Cloud Computing, in general, and SaaS, in particular, have been able to bring down costs mainly due to the widespread use of Open Source software in their platform stack. Vendors are able to transfer the cost savings they achieve with the use of Open Source software to customers. Open Source software can also add value in the SaaS model by building the confidence of the customers through mitigating the risk born by SaaS customers who have to put their data on the hands of third party vendors. In the eventuality that a SaaS vendor goes out of business, the customers have the option of getting their data out of the outgoing vendor and finding a compatible SaaS application or install the application in a cloud infrastructure and keep going since the source code of the application is available. This also provides more flexibility to the customer and reduces lock in.

Cloud computing offers a realization of SOA in which IT resources are offered as services that are more affordable, flexible and attractive to businesses.

The term cloud computing emerged in 2007 and refers to a technology which offers flexible dynamic IT infrastructures, QoS guaranteed computing environments and configurable software services. Due to its access on demand approach, it has been referred to as utility computing. Since its conception, numerous projects within industry and academia have emerged for example the RESERVOIR project – an IBM and European Union joint research initiative for Cloud computing, Amazon Elastic Compute Cloud, IBM's Blue Cloud, Microsoft's Azure scientific Cloud projects such as Nimbus and Stratus and recently Intel Corporation and Yahoo! Inc. announced the creation

[26] http://www.arenasolutions.com/about/index.html, www.cloudave.com/link/open-source-value-addition-in-saas http://www.zimbra.com/learn/

of a global, multi-data centre, open source Cloud computing test bed for industry, research and education[27].

(Wang & Laszewski, 2008), claim that there are still no widely accepted definitions for Cloud computing even though the Cloud computing concept has attracted much attention recently. They describe cloud computing as an infrastructure which permits users to build complex IT infrastructures by providing a platform which enable them to manage various software installations, configuration and updates. They argue that because computing resources and other hardware are prone to becoming outdated within a short time, outsourcing computing platforms in the cloud is a smart solution for users to handle complex IT infrastructures. (Wang & Laszewski, 2008), defined cloud computing as:

> A set of network enabled services, providing scalable, QoS guaranteed, normally personalized, inexpensive computing platforms on demand, which could be accessed in a simple and pervasive way.

National Institute of Standards and Technology (NIST)[28] provides an alternative definition of cloud computing. According to their definition:

> Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model promotes availability and is composed of five essential **characteristics,** three **service models** and four **deployment models**.

2.7.1 Functional aspects of cloud computing (service models)

Cloud computing offer users access to hardware, software and data resources, in an integrated computing platform as a service, in a transparent way. There various services offered can be categorised into:

[27] Global Cloud computing test bed http://www.hp.com/hpinfo/newsroom/press/2008/080729xa.htm

[28] http://csrc.nist.gov/groups/SNS/cloud-computing/

- **Software as a Service (SaaS):** SaaS is an emerging form of software offering with special characteristics. It involves Software or an application which is hosted in remote servers by a service provider and provided to customers as a service across the Internet. The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure (Mell & Grance, 2009). This mode eliminates the need to install and run the application on the customer's local computers. The applications are accessible from various client devices through thin client interfaces such as web browsers. SaaS therefore alleviates the customer's burden of software maintenance and support and reduces the up-front expense of software purchases, through less costly, on-demand pricing (Tucker, 2010). Salesforce was one of the first SaaS providers supplying enterprise resource software such as customer relationship management software (CRM) and also providing the cloud platform for building and running business apps. Other examples of SaaS include the Application Service Provider (ASP). The ASP provides subscriptions to software that is hosted or delivered over the Internet. Google's Chrome browser provides another approach to SaaS by adopting an open source model. As an open, modern software browser, it has the potential to improve their customers' cloud computing experiences. A new desktop could be offered, through which applications can be delivered (either locally or remotely) in addition to the traditional Web browsing experience[29].
- **Data as a Service (DaaS):** This refers to an offering which enables individuals and organisations to access Data in various formats and from multiple sources through the internet, the users can manipulate the remote data just as if it were on a local disk. Examples of this offering include Amazon Simple Storage Service (S3) which provides a simple Web services interface that can be used to store and retrieve, any amount of data, at any time, from anywhere on the Web. The DaaS providers enable their users to take advantage of the large storage resource and scalability offered by an external, cloud enabled data centre. DaaS could also be found at some popular IT services, such as Google Docs and Adobe Buzzword (Wang & Laszewski, 2008).
- **Infrastructure as a service (IaaS)** (Hardware as a Service (HaaS))**:** It refers to an offering which provides IT hardware, an entire data centre, network and processing provision and other fundamental computing resources to users on a pay – as – you - go basis (Tucker,

[29] Google Chrome http://www.google.com/chrome/,

2010). This enables the consumer to deploy and run arbitrary software, which can include operating systems and applications (Mell & Grance, 2009). The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems; storage, deployed applications, and possibly limited control of select networking components such as host firewalls. This form of computing has also been described as hardware as a service (HaaS). The offering is made possible by the rapid advances in hardware virtualization, IT automation and usage metering and pricing. Examples of HaaS offering could be found in Amazon EC2, IBM's Blue Cloud project, Nimbus and Eucalyptus.

Platform as a service (PaaS):

(Mckinsey and Company, 2008) report entitled *Emerging platform wars in enterprise software* described PaaS "as cloud based IDEs that not only incorporate traditional programming languages but include tools for mashup-based development". PaaS enable users to subscribe to their favourite computing platforms with customized requirements of hardware configuration, software installation and data access demands. Cloud platform services or *Platform as a Service (PaaS)* deliver a computing platform and/or solution stack as a service while using cloud infrastructure and sustaining cloud applications. It facilitates deployment of applications without the cost and complexity of buying and managing the underlying hardware and software layers (The Guardian, 2008). Examples of PaaS include SalesForce's Force.com[30] which is based on the SalesForce SaaS infrastructure and Apex language, Bungee Connect which is a visual development studio based on Java, LongJump which is based on Java/Eclipse[31] and Google App Engine (which is based on Python and Django). PaaS has the following key advantages:

- **Self service-** Tenants acquire and release cloud instances entirely through an automatic mechanism with little human intervention.
- **Cloud promotion-** The cloud is capable of cloning previous stages of an application which is still being developed, into new server capacity. This reduces complexity for the tenant and ensures consistency of configuration across stages.

[30] http://java.dzone.com/articles/what-platform-service-paas

[31] http://longjump.com/

- **Application decoupled from capacity-** the hardware capacity beneath an application can be easily adjusted as the business needs shift. This happens behind the scenes and can be done without the tenant intervention.
- **Better resource allocation-** because of the self service and decoupled nature of the cloud, the actual servers could be managed as pooled resources. Additional server capacity can be added when necessary and spare capacity can be better utilized where needed[32].

(Dubey, 2008), opine that while there are various other ways in which SaaS platforms provide value to users, a vendor must address one of three needs in order to create a viable offering:

a) application delivery: A run-time environment in which users can deliver their applications
b) application development: A development environment in which users could build applications
c) An avenue to participate in a broad ecosystem and reach a new market place of developers or users.

(Tucker, 2010), corroborates this by pointing out that most PaaS providers also offer marketing channels to developers in addition to their development platforms. Examples include Facebook.com.

(Smith, 2010), presents a list drivers and barriers to cloud computing adoption.

Table 2.2 Drivers of cloud computing technology adoption

[32] Platform as a Service (PaaS) Example http://www.google.com/search?q=example+of+PaaS&ie=utf-8&oe=utf-8&aq=t&rls=org.mozilla:fi:official&client=firefox-a

Scalability	Users have access to large amount of resources that scale based on demand.
Elasticity	The environment transparently manages a user's resource utilisation based on dynamically changing needs.
Costs	Pay-per-usage model permits organisations to only pay for resources which they need.
Mobility	Data can be accessed from anywhere at any time.
Virtualisation	Each user has a single view of available resources independently of how they are arranged in terms of physical devices.
Collaboration	Users can use the cloud as a way of working simultaneously on common data and information

Table 2.3 Barriers to adoption of cloud computing technology

Security	Security problems include data privacy and lack of control or knowledge about where data is been stored.
Interoperability	Lack of a universal set of standards and/ or interfaces. This results in significant risks of vendor lock-in.
Control	The amount of control users have over the cloud environment varies greatly between vendors.
Performance	The use of the internet as access medium introduces latency in communication.
Reliability	Many existing cloud infrastructure leverage commodity hardware which are known to fail unexpectedly.

2.7.2 OSS in the clouds

There are some compelling arguments for offering OSS on the clouds. The arguments take into consideration the unique characteristics of open source software, the advantages and challenges posed by cloud computing, the legal issues related to both cloud computing and OSS and also business and economic considerations.

Software attributes argument: OSS has had a difficult time breaking into the desktop environments despite its remarkable success in infrastructure and backend software systems. Typically the end user has found it difficult to take full advantage of the power of OSS hence it has been regarded as developer oriented or high end user software ((Krishnamurthy, 2003), (Stefan, Fabio, & Maria, 2007)). The advantage of having the source code available is minimal to an end user whose only interest in the software is to solve a practical life problem but it means so much more to a developer who may also be interested in optimising the functionality of the software or employ it as a module for other product(s). (Franke & Hippel, 2003), surveyed the motivations of webmasters who had adopted the Apache open source web server application, showed that the more skilled users who modified the source code were most satisfied with their decision. Building a PaaS infrastructure with OSS components offered to high end users therefore plays to the strengths of the software and meets the needs of a desirable user group.

Business arguments: Typically most OSS adopters and potential adopters have found issues with the mode of support services for OSS. The issue of continuity of the projects which maintain the products and hence the products themselves, has also been an important dilemma ((Calpy & Chenogorov, 2009), (Goode, 2004), (Seppä, 2006), (Karels, 2003)). While many companies have development models to meets the need for end user support (with varying quality of service), the PaaS idea of providing seamless support services, performance and quality assurance based on predefined service level agreements (SLA) seems a lot more compelling. Apart from meeting and important need for support, the PaaS option of providing OSS components to users also shifts the source of total cost of ownership (TCO) from capital expense (Capex) to operational expense (Opex). Since resources are paid for, on a pay as you go basis instead of heavy initial capital investments ((Buyya, Shin, Venugopal, Broberg, & Brandic, 2009), 2009, (Buyya, Ranjan, & Calheiros, 2009), (Tucker, 2010)). This is particularly important for SME and start up enterprises because unlike already established companies, they do not have huge sunk investments. In addition to this, the use of OSS components to provide such services may lead to a marked reduction in the cost of the services and quicker development time as opposed to if proprietary software is used (Riehle, 2007). It is also easier for a company to draw estimates and account for expenses made in

IT service subscription in this way rather than adopting software directly from the community without ever knowing how much will finally be spent on support and ancillary services.

Legal arguments: One of the major deterrents for enterprises to adopt OSS is linked with legal issues related to IPR ((Calpy & Chenogorov, 2009), (Karels, 2003), (Seppä, 2006)) A key characteristic with OSS licenses is that it requires the licensee to redistribute the source code of the software freely. However, most proprietary organisations, while wishing to benefit from OSS, will still desire to keep their source codes closed. The Open source licence (GPL v2) provides a loop hole which can and has been exploited by SaaS (. The GPL v3 unsuccessfully attempted to close the loop hole by including some provisions of the Affero General Public Licence (AGPL). However, this led to a clarification of the phrase "convey a work" by the Free Software Foundation (FSF) which states that:

> The mere interaction with a user through a computer network with no transfer of a copy is not conveying[33]

This statement effectively legalised the use of SaaS to offer OSS as a proprietary software service[34].

2.7.3 Cloud challenges

Cloud computing is faced with major challenges because as an emerging technology, many features and API's related to the clouds are still to be developed also standards still have to be established across different providers to ensure compatibility. Security of customer information, software and processes in the cloud is another major issue- It is important to command trust on a multi tenant platform. Business concerns such as reliability and vendor lock in- which may result due to restrictions in interoperability across different infrastructures, are important. There are also legal issues such as privacy, regulation compliance and third party involvement in a system hosted across distributed physical locations with different legal systems ((Tucker, 2010), (Wang & Laszewski, 2008), (Vouk, 2008)).

2.8 Summary

[33] www.gnu.org/licence/gpl3-final-rationale.pdf
GNU has no networked future http://www.linux-mag.com/id/3017

[34] Tim O'reilly claims here that the decision not to close the SaaS loop hole could not be made in GPL v3 because many SaaS providers had already incorporated OSS in their offerings.
http://radar.oreilly.com/archives/2007/07/the-gpl-and-sof-1.html

This chapter has reviewed the history of OSS, the evolution of open source systems and communities and different open source business models have also been examined. The legal issues related to software in general and OSS in particular have also been reviewed. The concepts of cloud computing have been examined in a bid to relate them to OSS. This has revealed some interesting possibilities and highlighted important challenges.

An in-depth literature review revealed that the current literature does not examine in-depth the possibilities of harnessing the power of OSS as a means to enable low cost and high quality SaaS service and leveraging the features of cloud technology to provide highly scalable on demand IT services. The next chapter will explore the diffusion of innovation theory through a literature review. The review on the diffusion of IT innovations includes micro level factors, network effect, telecommunications and IT industry level factors, and macro level factors. A framework for the aggregation and delivery of developer oriented open source software for high end users is developed and described.

Chapter Three

3. Diffusion of Innovation Theory

While open source software itself is not exactly a new innovation, the idea of providing OSS components in a cloud service platform is a relatively new proposition. The idea of SaaS was initiated by Marc Benioff of salesforce.com in 1999. Software as a service has its roots in the application service provider (ASP) model. ASPs "provide a contractual service offering to deploy, host, manage, and rent access to an application from a centrally managed facility at a low cost. According to research firm IDC[35], although it showed much promise, the ASP model never experienced widespread adoption. Software vendors continued to require clients to purchase comprehensive applications, despite the fact that most only need a fraction of the functionality. In addition, few ASPs changed their licensing models to reflect the evolution of software delivered as a service.

ASP has slowly evolved into software as a service or "on demand," as the technology and services offered by SaaS providers have progressed and advanced ((Pettey, 2006), (benilian, Hess, & Buxmann, 2009)). According to IDC, the key characteristics of software as a service include:

- Network-based access to, and management of, commercially available software
- Activities that are managed from central locations rather than at each customer's site, enabling customers to access applications remotely via the Web Application delivery that typically is closer to a one-to-many model or multi-tenant architecture.

Software as a service vendors are able to compartmentalize functionality into practical module applications, so that organizations only pay for what they use. The current state of software as a service is driven primarily by cost, convenience and time savings on the part of the customer (Kern, Lacity, & LWillocks, 2002), While on the part of the provider, the important proponents are: defence against new entrants into an ISV's space, a means to expand into new markets in order to

[35] [NCB, 1999] NCB, "Businesses urged to tap E-Commerce potential more aggressively". National Computer Board, Singapore, 4 June 1999. (Available: http://www.ncb.gov.sg/ncb/press/040699.asp)

grow revenue and increase market valuation. For the established ISV, SaaS is about “disruptive” innovation; appealing to a new set of customers who value different attributes and redefining the price/ features ratios for existing customers[36]. Further drivers of adoption in the SaaS model include increased flexibility, customization, and configurability for specific business or market conditions. Key market drivers will evolve from today’s cost-effective software management solutions to enabling companies to transform their business structures and processes ((Koenig, 2006), (Turner, Budgen, & Brereton, 2003), (Metz, Eschinger, Pang, & Dharmasthira, 2008)).

Major inhibitors to SaaS adoption by enterprises include security concerns - compliance with data privacy regulations - , reliability and continuity - making sure that services are available and accessible when needed -, and process dependence -meeting agreed service quality level- (benilian, Hess, & Buxmann, 2009). A wide category of applications have been offered as SaaS ranging from Office and collaboration to CRM and ERP applications by companies such as Salesforce.com, yahoo and SAP with different levels of success. (T-Systems, 2008), points out that SaaS is not ideal for every category of application.

This study considers the offering of OSS as a service model to be a very contemporary business model therefore, in order to explore the subject in detail, a review of salient literature on the diffusion of innovations (DOI) theory will be used to gain understanding on how the DOI theory influences the implementation of a software service provision business model based open source software.

Since the topic is very much dealing with implementation of new innovation, we choose to review the literature on the diffusion of innovations (DOI) theory to help us understand how the DOI theory influences on the implementation of a software service provision based open source business model.

Different literature on technology innovation diffusion theory provides diverse views on new technology adoption. While in some instances, simple theories such as the epidemic effect premises (innovations are assumed to spread like diseases when potential users are in contact with the technology users whom they learn about the new technology and immediately adopt it) are used to

[36] SaaS transformation demystified and accelerated. http://www.oracle.com/technology/tech/saas/case-studies/cscprofile.pdf

explain the gradual, typically S-shaped diffusion of technologies over time ((Koski, 2008), (Griliches, 1957)), they are clearly insufficient in explaining the adoption and diffusion dynamics of complex knowledge intensive technologies such as IT. Other theories, such as Game theory (Canepa & Stoneman, 2004) cooperation theory ((Axelrod, 1984); (Axelrod & Keohane, 1986)) and transaction cost theory (Williamson, 1975), has been used to examine OSS delivery as SaaS. Due to the constraints of time and scope of the research study (i.e. M. Sc level research study), In-depth theoretical review is restricted to only one background theory. Reference is also made to the network effect theory which suggests that as the number of users of the technology increases, so too does the benefit of the adopting firm ((Koski & Kretschmer, 2007), (Saloner & Shepard, 1995)).

In (Rogers, 1983), diffusion is defined as the process by which an innovation is communicated through certain channels over time among the members of a social system. The process of providing OSS as a service to high end users typically involves both new technologies such as a PaaS or cloud infrastructure or requires users to adopt new approaches to their work. Based on the scale of innovation efforts, DOI theory can be classified into three groups: micro level factors, industry level factors, and macro level factors (Nahar, 2001).

3.1 Micro level factors

(Carr, 1999), describes micro level theories as those that focus on the individual adopters and a specific innovation or product rather than on large-scale change. At this level, the diffusion patens among related organisations and individuals are brought to light. The scope of diffusion in this case is restricted to a narrow and well defined locus. Technology diffusion research has focused on five factors: the characteristics of the innovation - which includes; Relative advantage , compatibility , trialability, observability , and complexity of the innovation - ,the technology supplier, the means by which adopters learn about and are persuaded to adopt the technology, the decision-making process that occurs when individuals consider adopting a new idea, product or practice, the consequences for individuals and society of adopting an innovation and the characteristics of the technology adopter ((Rogers, 1983), (Fichman, 1992), (Nahar, 2001)).

In this respect, the theory explains how the nature of the particular OSS component and integrated platform services impact the decision making process of adopters and encourage implementation and further diffusion. Micro level theories take into consideration the unique strengths and weaknesses of the innovation to determine the challenges which it poses to the individual “technology supplier”- provider organisation- and the perceived benefits to the “technology recipient” – high end user group-.

3.2 Macro level (Systemic Change) theory

(Carr, 1999), defines macro-level theories as those that focus on the institution and systemic change initiatives. Systemic change theories, typically involve the adoption a wide range of innovative technologies and practices. They have a broad scope and usually engross broad aspects of curriculum and instruction. The implementation of OSS as a service business model depends on some industry level specific factors and the general prevailing atmosphere of the target market.

(Nahar, 2001), sited quality of IT education as a major factor affecting the ability to implement and adopt information technology. (Fichman, 1992), points out that, some technologies cannot be adopted as a “black-box" solution, but rather, impose a substantial knowledge burden on would be adopters. Since information technology is considered to be a knowledge intensive technology which is perpetually evolving, it requires substantial and high quality adaptive learning (Lai & Mahapatra, 1997). Important macro level factors to consider in the context of this study include: economic, legal, political and socio cultural factors.

3.3 Organisational level factors

In the case where the adopter of the technology is an organisation, the dynamics leading to an adoption decision being made are very different and most of the classical diffusion theories do not fit. Therefore extensions are made by reviewing other theories which bear more closely with this circumstance. Unlike in individual adoption scenario, organizational adoption of an innovation is not typically a binary event, rather, it involves lots of politicking over time. The organizational decision process, especially when there is no dominant individual decision maker, involves complex interactions between vested stakeholders (Andrew, Harold & Poole, 2000). The decision to adopt an innovation in an organisation is an active and dynamic process, the persons and settings involved are not rational players, and the advantages or disadvantages of the innovation are spread unequally among those involved (Scott, Plotnikoff, Karunamuni, Bize & Rodgers, 2008). Decision makers in organisations bring with them interests, values, and power that further shape and add complexity to the innovation adoption process. The resultant decision taken reflects not only the characteristics of the technology itself, but also a compromise among conflicting interests and forces.

3.4 Conclusion

By exploring the theoretical framework of diffusion of innovations, valuable insight is obtained about possible source and pattern of diffusion. Many of the insights gained from cumulative research on diffusion of innovation are applicable to adoption of OSS as a service. However, the multitude drivers of innovation suggested in the literature raise the question concerning which drivers of innovation are of particular importance when considering the provision of OSS as a service and whether unique drivers of an innovation can be identified in this context. (Fichman, 1992), points out that the opportunities to apply classical diffusion without modification are very rare in the context of IT adoption because of the complex factors involved. Complex IT innovations are not stand-alone and independent implementation is very unlikely. The extremely networked nature of IT and its dependence on industry wide standards requires collaboration among different firms and a common infrastructure (Koski & Kretschmer, 2007).

(Bonaccorsi & Cristina, 2003), concluded, on the basis of reviewed literature that the adoption of OSS and its diffusion are influenced by the perceived intrinsic value of the open source software, the negative externality effect as a result of the other more dominant standard, the positive externality effect as a result of association with OSS communities and the competitive reaction from the proprietary software firms.

Chapter Four

4. Conceptual Framework

In this chapter, a conceptual framework is developed for creating and implementing a PaaS business model which is based on the aggregation of relevant OSS components and establishment of an on demand service platform for software development firms in specific industry. This study distinguishes between a business model and a business process model. Although these two concepts are quite similar and are often used interchangeably, they do have some subtle differences: While business model refers to a firm’s logic for creating and commercializing value, a business process model is related to the way in which a business case is implemented in processes ((Gordijn, Akkermans & Vliet, 2000), (Gordijn & Akkermans, 2001), (Osterwalder, 2004)).

According to (Miller, 1987), the goals of the conceptual framework are, to describe existing practice, prescribe future practice; and finally, define key terms and fundamental issues. It provides the basis for future debate especially in relation to prescriptions for future practice and a broad definition of key terms and fundamental issues. In this research, the framework is intended as a tool to enable providers of OSS on a PaaS business model to identify what kind of OSS components are feasible for this kind of software service provision and consequently how to relate with the various open source communities which develop the identified components. It brings into light the challenges involved in building such a platform, who the target users (clients) are, their readiness to adopt such services and how to manage the provider-customer relationship. The framework provides structured ways in which various process planning and control can be efficiently and effectively managed while calling attention to some of the limitations of the model.

The framework is based on the literature review as well as a proposal for conceptual framework for business model research by (Lambert, 2008), and business model ontology by (Osterwalder, 2004).

This framework includes nine generic elements that provide the foundation for the business model and process representations of OSS provision on a PaaS.

Figure 4.1 illustrates an adaptation of (Osterwalder, 2004), business model ontology. The first tier represents the external resources which are pulled by the business in order to build an offering while the second tier represents the core activities and resources of the business which are combined with the external resources to build a value proposition which is then targeted towards the satisfying the needs of a designated market. The third level explains the revenue streams, pricing strategy and cost structure of the model. Experience and knowledge is gained through practice and used to improve the proposition and the business processes. The performance of the firm in the real world determines both the robustness of the model and the competence of the management team which implements it.

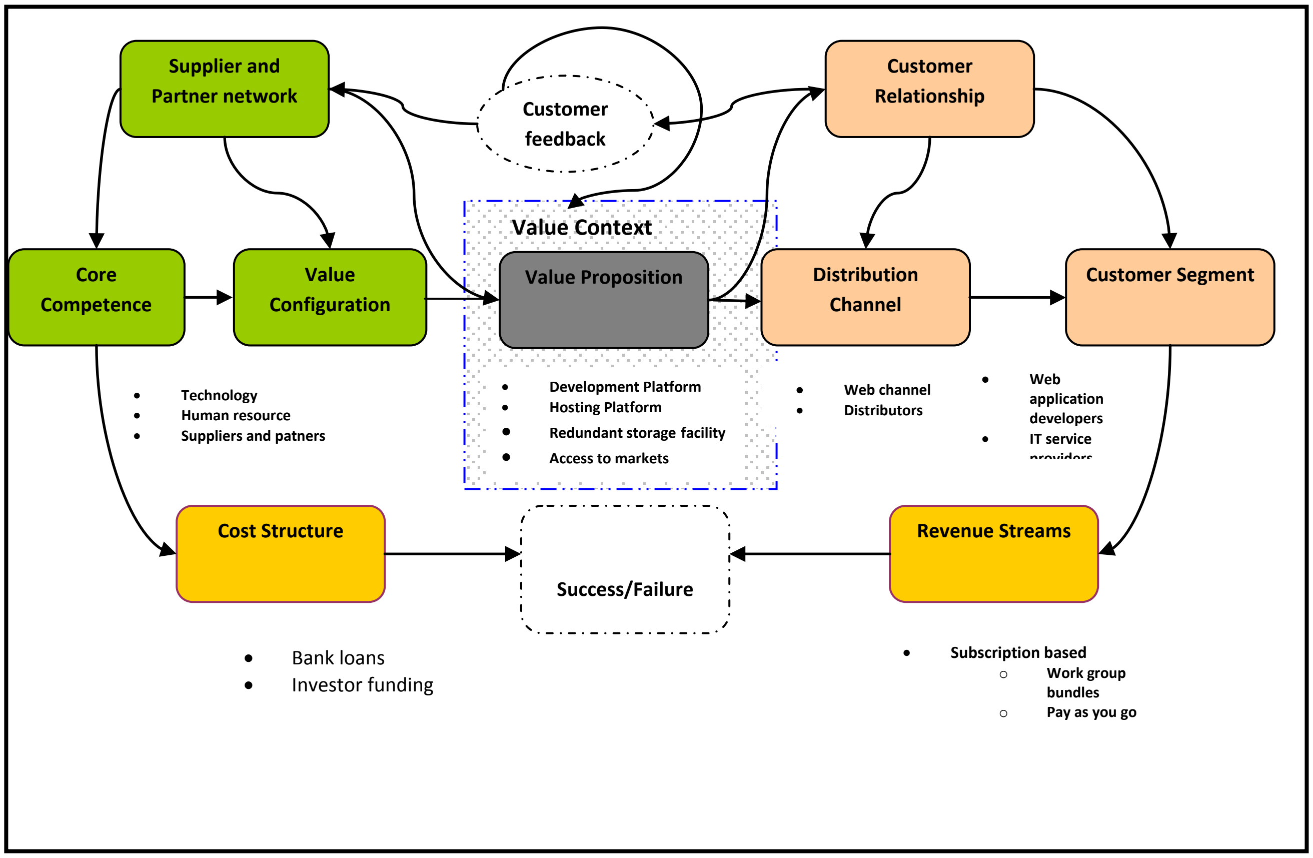


Figure 4.1 PaaS Business model representation

Source: Adapted from (Osterwalder, 2004)

4.1 Definition of OSS on PaaS business model

Business model means different things to different people (Linder et al 2000). For the purpose of this study, business model is considered as a set a conceptual tools containing a set of objects, concepts and their relationships, which have as objective, to express the business logic of a specific organisation. Understanding the concepts and relationships of a business model facilitate the abstraction and representation of the value provided to customers, means of provision and the financial consequences of the endeavour.

In this sense, a business model is concerned with providing information that reflects the nature and method of product, service and information flows in a structured manner which brings to light the roles played by different actors and the strategic choices they have to make in order to satisfy specific user needs and ensure economic sustainability of the entity. ((Timmers, 1998), (Alt & Zimmermann, 2001), (Gordijn;Osterwalder;& Pigneur, 2005))

The offering of OSS components on a “platform as a service” basis, requires the provider to make a decision on exactly what services to offer – different combinations of PaaS offerings may be proposed by a provider to support the application development lifecycle and delivery of SOA web applications and services through the internet. A comprehensive PaaS offering provides all service options in an integrated development environment within the actual target delivery platform, with source code control, version control, dynamic (interactive) multiple user testing, roll out and roll back with the ability to audit and track who made what changes when and to accomplish what purpose. In PaaS, the computing layers are transparent and available to the programmer thus facilitating deployment of applications without the cost and complexity of buying and managing the underlying hardware and software. Many PaaS offerings include workflow facilities for application design, application development, testing, deployment and hosting as well as application services such as team collaboration, web service integration and marshalling, database integration, security, scalability, storage, persistence, state management, application versioning, application instrumentation and developer community facilitation; these adjacent services are provisioned as an integrated solution over the web. In defining the business model, a strategic analysis is made and business goals which represent what the firm hopes to achieve, in terms of quality, sphere of influence, volume of revenue and time frame are set. The setting of goals makes it possible for the business to define objectives, identify executable functions and design relevant strategies to meet these objectives. Understanding the internal capabilities of the organisation and surrounding business and IT ecosystem is an important starting point to setting clear and realistic business objectives. In setting these objectives, the organisation should be able to make informed projections into the future of the industry taking into considerations both the articulated and unarticulated needs of all the stakeholders in order or be competitive (Herselman;Botha;& Brits, 2007).

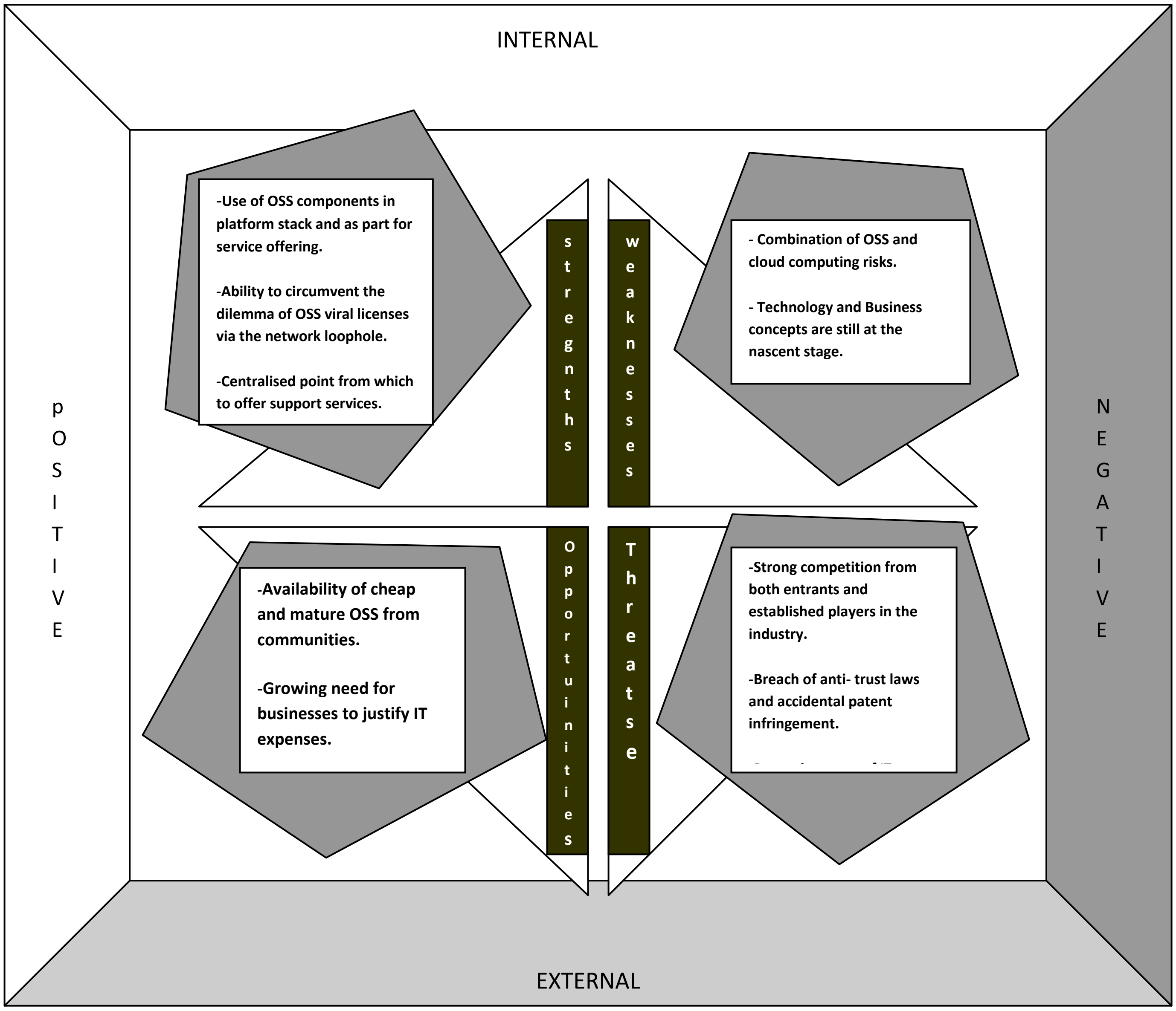

INTERNAL
pOSITIVE
NEGATIVE
EXTERNAL
strengths
weaknesses
Opportunities
Threatse
-Use of OSS components in platform stack and as part for service offering.
-Ability to circumvent the dilemma of OSS viral licenses via the network loophole.
-Centralised point from which to offer support services.
- Combination of OSS and cloud computing risks.
- Technology and Business concepts are still at the nascent stage.
-Availability of cheap and mature OSS from communities.
-Growing need for businesses to justify IT expenses.
-Strong competition from both entrants and established players in the industry.
-Breach of anti- trust laws and accidental patent infringement.

Figure 4.2 SWOT Analysis of OSS PaaS business model

4.2 Strategic relationship with suppliers and partners

After the PaaS provider has determined what sort of services to offer, it needs to investigate the sources of the requisite elements to enable it meet its service objectives. The provider will have to build strong relationships with relevant OSS communities in order to have first hand information about the direction of their projects, provide feedback from its customers, manage service support and maintenance and navigate the community and software towards its goals. This may not be an easy task considering the communities may themselves have their own objectives which may not necessarily be the same as that of the PaaS provider. The solution to this problem may be that the provider participates actively in the community and builds enough influence to be able to persuade the community to streamline their objectives with the provider's. Other incentives could also be provided to motivate OSS communities such as access to some of the provider's platform services. The best possible outcome is realised when the OSS community and the platform provider are merged so that the community can be viewed as part of the provider's ecosystem.

It is also vital for the PaaS provider to establish partnerships with both online and offline partners in order to expand their market reach and improve their service offering. In the course of building strategic relationships, the provider must define its role in an ecosystem and sketch out its interaction with other players in the business environment. Category of partners for the PaaS vendor may include in addition to open source communities, other SaaS vendors, cloud-savvy IT Service companies, providers of online business services, web application developers and established brands that are extending their reach onto the Web. By understanding the rules which govern relationships among web based businesses, the PaaS provider can obtain maximum benefit from such partnerships and also learn to stay competitive.

4.3 Value added process

The value added process relates to how the firm uses its unique characteristics to distinguish itself and its offering from other firms in the industry. The value added process blends together the resources, capabilities and activities of the firm to gives it distinct advantages over competitors by improving on the fundamental value proposition of the business. The goal of the value added process is to produce an end product or service whose value exceeds the cost of producing the product or providing the service. The key elements of a PaaS value added process include components such as; technological and knowledge resources, raw materials, labors force, storage, logistics, organizational structure and strategy, overhead costs. A combination of these elements represents the businesses core competence.

Technological resources: The PaaS provider should have the technology to manage infrastructural concerns such as service availability, load balancing, scalability, system backups, operating system patches, and security and also provide customizable services which can support multiple devices and are easy to upgrade and integrate with other applications. A PaaS provider could provide the tools for building a rich development environment which include databases, integration tools, user interfaces, asset management and bug tracking features. An example of a PaaS stack from Force.com reveals the key layers of technology and service which make up the platform.

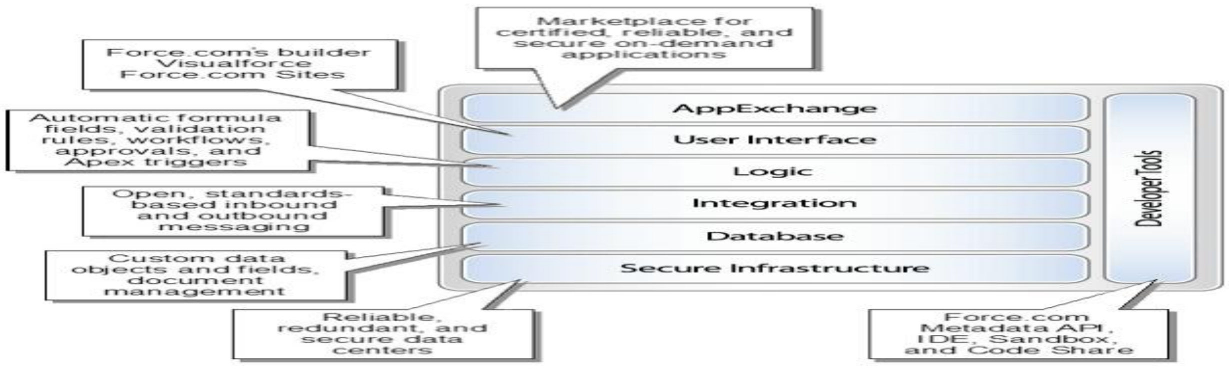

Marketplace for certified, reliable, and secure on-demand applications
Force.com's builder Visualforce Force.com Sites
Automatic formula fields, validation rules, workflows, approvals, and Apex triggers
Open, standards-based inbound and outbound messaging
Custom data objects and fields, document management
AppExchange
User Interface
Logic
Integration
Database
Secure Infrastructure
Developer Tools
Reliable, redundant, and secure data centers
Force.com Metadata API, IDE, Sandbox, and Code Share

Figure 4.3 PaaS technology and service stack

Source: (force.com cloud platform, 2009)

Two key technology enablers for platform as a service are Multi-tenancy and Metadata. Multitenant infrastructures enable greater cost-efficiency than their single tenant counterparts (the ASP model) by making it possible to share a single physical instance and version of an application to multiple users in such a way that individual users have virtual isolation from one another and can use and customize an application as though they each have sole access and control over it. On the other hand, using runtime engine which generates application components from metadata makes it possible for tenant specific data to be kept secured in a shared database, make patches and upgrades to core application code base without affecting tenant specific customization and enable individual tenants to customise their application interface and business logic in real time without affecting the functionality or availability of the application for all other tenants. Other technological competencies include: Billing, metering and monitoring services, run-time environment, on demand infrastructure, remote infrastructure and physical data centre.

Knowledge resources: the IT industry is a very knowledge intensive industry. For any IT based business to thrive it needs to have a strategy of recruiting, developing and managing knowledge resources. Possessing vital knowledge in an industry could be an important tool against competition and can provide added value to the value proposition.

Competent and experienced labour force: Ultimately the quality of service being offered to the customer and the direction and dimension of growth achieved by the business depend on the ability and experience of the human resource possessed by the firm. A good value proposition could under perform in the market and growth opportunities lost if the people behind the offering are not competent or experienced enough.

Organisation structure and strategy: The organisational structure of a business can be a key source of value added resources. In the case of a PaaS provider which works in tandem with essential OSS communities, they can use the advantage of having a large and supplier base to develop production efficiencies and dramatically reduce both cost of production and development cycle of their offerings. It also provides a vast pool of talents and knowledge which can also be used to test new innovations before release to the general market. In the very dynamic and rapidly growing IT industry, it is necessary for the firm to be flexible and adaptable so as to respond quickly to changes in the industry, manage rapid growth in both customer populations and demands, seize opportunities and discontinue policies which do not work.

4.3.1 Context design

Context in a service based business model such as platform as a service is defined by the value requirement designed into the core service or product that the value-added services fulfil (Hamari & Heikkilä, 2010).

According to (Hamari & Heikkilä, 2010), traditional authorities in marketing assume that the context in which products are offered is defined exogenously, outside the sphere of influence of the service provider. However they contend that just like firms selling virtual goods who create and design the rules (context) and mechanics that determine the activities and specific needs of the participants in the context of the virtual world, traditional business can also deliberately design rules and values around their core proposition which enables them to create pre-determined sets of segments into which users self-select themselves.

Businesses can play an active role in shaping the environment in which their products and/or services are offered by creating both vertical and horizontal options for the user and influence the manner in which goods and services are consumed. This ability to create parallel markets along the main offering by creating needs and providing solutions result in additional value to the core proposition.

(Bouwman, Vos & Haaker, 2008), points out that understanding the context is a crucial starting point for formulating intended value (designing products and services). The value configuration process is iterative and intertwined. In the platform as a service business model, each platform provider can design rules and methods to facilitate and improve customer’s productivity and efficiency while increasing their investment and usage of the platform and thereby constituting an important part of the overall revenue generation mechanism. By differentiating with additional features and services such as access to libraries of OSS, access to communities of developers and relevant forums, provision of certification for users’ applications and an opportunity to advertise and gain recognition among peers and other community users based on such parameters as level of activity, commitment and quality of service within the ecosystem. Different pricing strategies are enabled according to different customer characteristics and needs. Some customers may only wish to have their applications deployed and hosted on the platform while others may want to use all the services of the platform. Where the design of the context takes the objectives of the customers into considerations, it contributes to the motivations for designing more of the platform services.

According to (Hamari & Heikkila, 2010), the context component of the business model can be regarded as part of the total value proposition because it is defined by design aspects of the service. And therefore it is located under the concept of product innovation (a product/service pillar in the business model framework). The context is purposefully designed to create value for the value offering and hence create further demand for the products or services.

By offering an avenue and establishing rules on how users interact with the platform, the platform provider designs a context which permits both horizontal and vertical segmentation (based on the customer's intention of usage). The vertical axis depicts product or service quality which scales from entry level to professional versions while the horizontal segmentation provides different content dimensions and complimentary products or services. In addition, the service provider can enforce the customer to move through the all vertical steps, hence inducing service consumption behaviour on each step and encourage the user to invest more in the platform. The more invested a user get into the services by acquiring skill on the different service offerings, establishing their reputation among the associated developer communities patronised by the platform provider, the more locked- in they become.

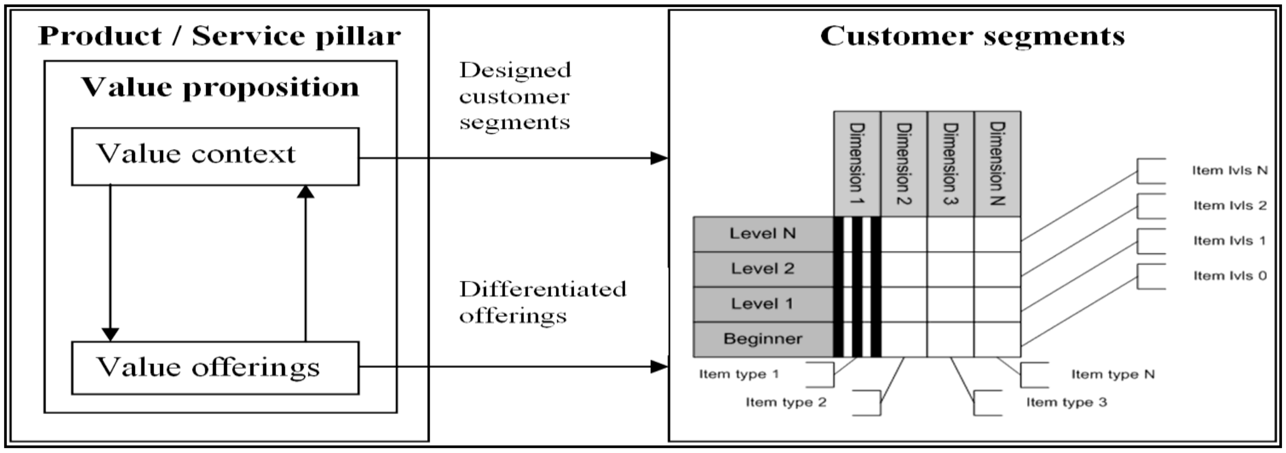


Figure 4.4 Role of value context in defining customer segments

Source: (Hamari & Heikkila, 2010)

4.4 Value proposition

This is the centre piece of the business model. The value proposition represents the object of value which the business offers to its customers and also the benefits which the customer perceives from the offering. A value proposition can be a product or service offering. It is the main reason why the business exists and describes the way a business differentiates itself from its competitors ((Osterwalder, 2004), (Lambert, 2008)).

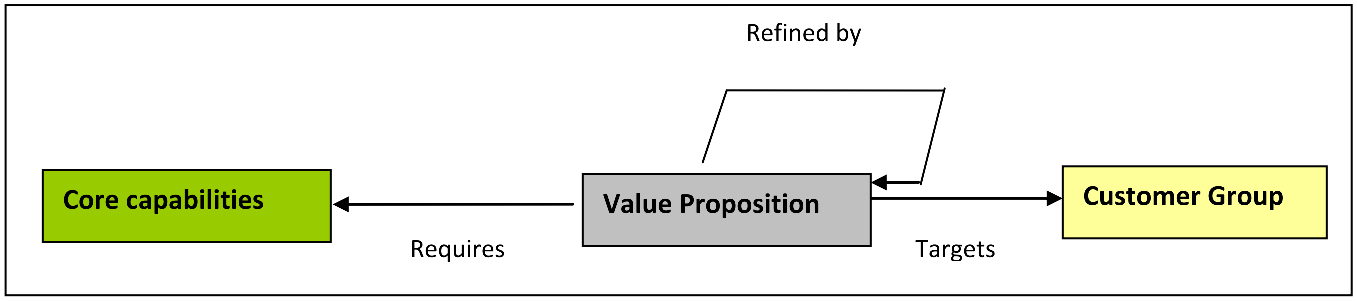


Figure 4.5 Relationship between the value proposition and other elements of the business model

Based on data collected from the analysis and decision phase, the PaaS provider identifies its business objectives and then breaks them down into meaningful process objectives that pertain to the service(s) the provider is offering. The core value proposition could be broken down to any of, or a combination of some or all of the following services[37]:

- **Provision of an integrated development Environment:**
  - **Add-on development facilities:** They allow customization of existing SaaS applications, and in some ways is the equivalent of macro language customization facilities provided with packaged software applications such as Lotus Notes, or Microsoft Word. Often these require PaaS developers and their users to purchase subscriptions to the co-resident SaaS application.
  - **Stand alone development environments:** These types of PaaS environments do not include technical, licensing or financial dependencies on specific SaaS applications or web services, and are intended to provide a generalized development environment. The developer builds entire new applications using the platform provider's own on-demand development infrastructure, tools and collaborative development environment. The developers do not have to worry about initial start-up investments in software and hardware infrastructure. This permits the customer to focus on their core competences while the PaaS provider takes care of security issues, uptime, back up and upgrades. The type of application they build is not constrained, but the infrastructure choices are limited to those which the PaaS provider offers. The trade-off for the customer is that they may be tied to a particular platform, with no easy

[37] http://www.cloud-y.com/index.php/paas-cloud-platforms/24-platform-as-a-service?format=pdf
http://knol.google.com/k/alfonso-guti%C3%A9rrez/what-is-paas/294ccfj1s1mhd/2#

way of transferring their application elsewhere should it become necessary. The use of web 2.0 technologies gives the users a near desktop like experience.

- **Provision of Run time Environment:** It is not a comprehensive PaaS offering as it only provides hosting-level services such as security and on-demand scalability and lacks development, debugging and test capabilities. However, the PaaS provider offers global access to the customer, any time and on multiple user devices. Some providers such as OpSource, also offer additional SaaS-specific services around the core hosting service such as an integration bus.
- **Open Platform as a Service:** Unlike other types of platforms which restrict the developer to specific languages which they support, the open platform lets the developer use any programming language, database, operating system, and server that they favour. The provider will have to decide not only on how to facilitate the developers deploying their applications in the cloud but also the ease with which they can move it from one cloud to another should the need arise. This creates openness in the platform and reduces lock in. Depending on the quality of the provider's services, this attribute can be pitched to ward off competition or make the model vulnerable to competition.
- The PaaS provider may also use OSS components in their platform stack and also offer pre-packaged OSS components which enable them to offer cost efficient pricing strategies as a result of gains realised in production efficiency.
- **Access to Market:** The PaaS provider offers ISVs, IT service vendors and enterprise customers a means to participate in a broader business environment and expand into new markets of users and developers

4.5 Distribution channel

This refers to the means through which the business delivers their offering to the market. PaaS is a web based service and as such can be accessed directly from the web. The PaaS provider's web site is a primary channel for the provision of its services to customers. Additional channels include third party distributors such as Daiwabo Information System Co., Ltd[38]. The primary challenge of the

[38] Daiwabo Information Systems (DIS) offers the following services: Facilitates search of SaaS/PaaS products and place orders, perform procedures such as order management and cancellation on-screen and obtain services for partial use of SaaS/PaaS products on a trial basis: http://www.pc-daiwabo.co.jp/english/release/091207.html

distribution channel is to increase awareness of the offering, extend its reach to relevant market. Awareness can be enhanced by OSS communities with which the firm has a relationship, SaaS vendors who develop or deploy their software on the provider's platform, other software developers and partners of the firm.

4.5.1 Defining target market

It is important to define a precise target market even if it is only a tentative delineation for the initial phase of the business. This enables the business to set realistic objectives and provides a reasonable template from which the broader market can be extrapolated. The PaaS provider needs to know exactly what their customers are interested in and have an idea of the potential market size in terms of possible number of customers. They also need to make projections about the target market in terms of how much of their service(s) the customer might subscribe for, and how they might be affected by trends and policies. In setting a market share objective, specific questions that are directly related to the service will have to be answered. Such as:

- The typical customer's income (capital).
- Typical customer organisation's size (employee size, geographic spread, market).
- Customer's business orientation (the basic offer addressed by a customer to its customers), preferences and practice.
- Potential market size for this segment (this seeks to determine the possible number of prospective customers in the niche and gives an indication of the maximum possible revenue obtainable from the market).
- Another key aspect to consider when selecting the market segment is how quickly it responds to changing trends in the general market and how strongly it is influenced by external network factors, technology innovation and globalisation.

In any case, the major challenge is to establish general customer needs and identify specific customers who are likely to experience these needs. Segmentation is determined by matching the benefits provided by the offering and the need of the prospective customers. In the case of PaaS customers, these needs may be grouped into Strategic, operational and functional needs.

Table 4.1 Categories of customer needs

Strategic Needs	**Operational Needs**	**Functional Needs**

-Increasing Market size and revenue base -Minimising competition - meeting established goals, visions and values	- Reducing development life cycle - Reducing downtime, operating cost and improving productivity -Reducing time to market -Improving Product and service quality - promoting and facilitating innovation. - Internal efficiency e.g.: providing corporate wide communication system	-Data storage needs -Security - Ensuring availability of service (by implementing clustering across multiple physical servers). - Scalability (remedied by dynamically resizing clusters and/or migrate live cluster nodes to different physical servers)

Market definition facilitates the preparation of marketing promotions and allows marketing and sales programs to focus on the subset of prospects which are most likely to purchase the offering. It also ensures high return on marketing and sales expenditure.

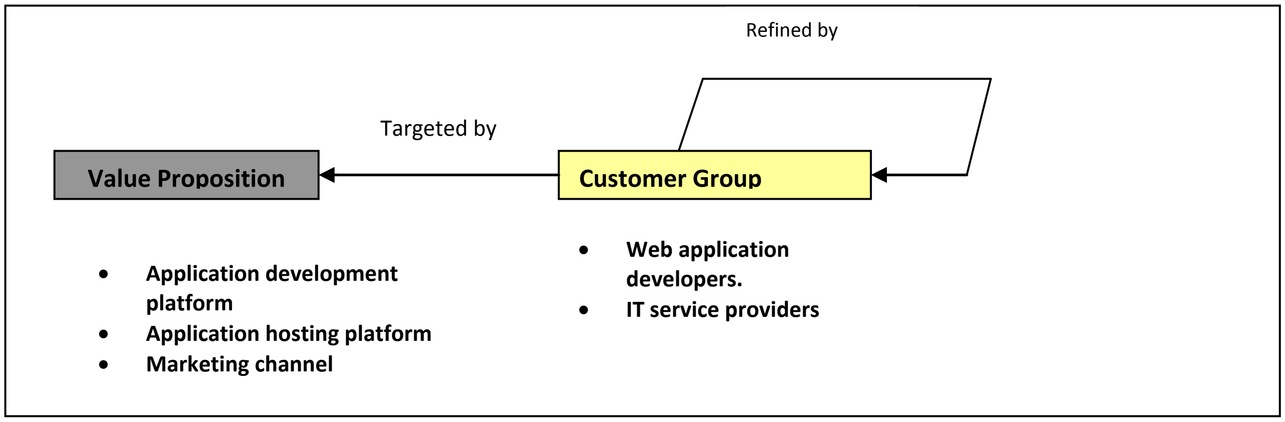


Figure 4.6 Relationship between value proposition and customer segment

4.6 Customer relationship

Customer relationship is a business philosophy which aims at finding ways to keep the customer satisfied and loyal to the business's proposition. Customer relationship management activities are driven by the needs of the customer. In a subscription model it is essential for the value proposition to be design from a customer centric point of view because the manner in which the customer consumes the offering and the revenue logic (the Euro value of a sale, the length of the sales cycle and the impact of making or losing a subscription are much more significant than for a retail business). For this reason, a successful PaaS business model requires in-depth planning and analysis not just of the product, placement, pricing and promotion, but also of the customer. There is a strong need to build trust and understanding between the PaaS provider, the customer, the suppliers and partners of the business. Important activities to carry out include:

Identifying the ideal customer and having a clear profile: characteristics of the model customer such as their specific needs, business orientation, firm size and preferred billing option should be collated and stored.

Focus on existing customers: While it is important for marketing and sales strategies to aim at expanding their reach to new customer prospects, it is always necessary not to forget the needs of existing customers. Acquisition of new customers is often more expensive that retaining existing customers.

Address customer complaints: It is important to follow up on the customer's complaints, resolve them as quickly and efficiently as possible, inform all persons in the organisation who need to know about it where necessary, offer training to personnel so as to avoid a repeat of the problem.

Develop and apply appropriate metrics: metrics could be designed to measure profitability and identify patterns in customer behaviour and preferences. Appropriate customer relations management software applications should also be used to facilitate the process and make it efficient.

Test the system regularly: regular and random test calls may be made to the customer contact persons in the firm (call centres and project managers), in order to evaluate the efficiency and effectiveness of the system.

Ultimately, having a successful customer relationship management program depends on the strategy been used, the ability of the personnel implementing the strategy and the technological and management tools they have.

4.7 Revenue streams and pricing strategy

In the “as a Service” paradigm, the change from offering “products” to “services”, from “acquiring” to “subscription” implies the need for defining the best way of charging for the solutions offered. The PaaS providers have to decide on the most effective and efficient way of fixing the right price to their offering. They have a number of different alternatives and factors to consider when choosing their price strategy. Most PaaS and SaaS providers today typically use these options:

- Charge on a temporal basis: This means charging the customers on a regular basis (usually monthly).
- Charge per user: This is a very popular billing method. The customer pays per computing node.
- Pay for the resources: This refers to computing resources such as, Compute usage per hour, data transfer in/out per GB, IO requests per million, storage per GB, data transfer in/out to storage per GB. This pricing structure is common in IaaS and PaaS offerings (example include Amazon Web Services)
- Pay for the features: The customer is charged just for the features in the solution which they use. This is common in PaaS offerings.

Generally the on demand model for software product/service vending offers a more flexible and less risky alternative to the traditional license-based on-premise software. However, in selecting an “as a Service” option, the provider must keep certain objectives in sight if they hope to sustain a profitable business model:

- In order to increase uptake of their offering and win new customers, the pricing strategy has to be interesting for new customers. This may be achieved by having a free or trial version or simply a ‘pay-as-you-go’ strategy which starts cheap and increasingly become more expensive. However, due to its inherent uncertainty attributes, the “pay-as-you-go” option may not seem very attractive for enterprises and organisations that would like to be able to predict the total cost of the services before committing themselves.

- Therefore the next challenge for the provider is to offer their users an option which enables them to make fair predictions of their costs. Coghead had an interesting strategy where they offered workgroup bundles (with discounts) and a more flexible 'pay-as-you-go' model. Users were allowed to choose a preferred billing option depending on their needs. There were four different workgroup bundles: plus, pro, premium, business, each one with a fixed price for a certain number users/records/space. Opting for a bundle was cheaper than having the same amount of usage via 'pay-as-you-go' and the price paid per bundle and what it encompasses of were predictable (Malik, 2006).
- Another objective is to scale the customers' commitment once they start using the system. This is achieved by offering a combination of different features and resources which make the user feels they are getting more value from the system by spending more.
- The provider also has to bear in mind that a trade-off of providing all these different billing strategies is a problem of increased complexity. This is a common problem in PaaS and SaaS offerings, and this can slow their adoption by the market.
- Finally a good pricing strategy ensures that the customer does not abuse in the use of the solution by being able to go around billing conditions and access and use services and resources for free or circumventing full cost.

4.9 Summary

This chapter has examined the OSS on a PaaS business model and has also identified important elements in the model which need to be properly addressed with well defined objectives for the model to succeed. The major actors in the model are the suppliers of software components which are used both in the platform stack and as part of the service offering, partners with whom the company liaises in other to add value to its proposition, the platform provider and their customers. The value proposition is one of the most important elements of the business model. In this case, there is a strong network effect. The more users adopt the service, the richer and stronger the platform becomes. Segmentation of the market permits the platform provider to focus their activities (promotional and sales) and competences (services, technology, R&D) to providing better quality services at a cheaper cost to the targets who can benefit most from it. The selected revenue collection scheme is almost as important as the product or service offering itself and understanding the nature of the customer and the service enables the provider to determine the best revenue and pricing strategy. Ultimately, the success and failure of the model will be determined by its suitability for the chosen market in terms of sufficient demand, available technology enablers, complementary products and services and the availability of competent labour resources.

Chapter Five

5. Research Methodology

This chapter describes the modality for selecting and employing the research methods within this study. It aims at enhancing an understanding of how the research was conducted and how the study is structured. The chapter seeks to; (a) provide a background to the choice of research methods; (b) describe the selection of research methods and elaborate the research design; (c) explain the data collection; and discuss the validity of the research.

5.1 Logic behind the choice of research method

Choosing an appropriate methodology for the research is fundamental to its successful execution. The choice of research method has been done so that it addresses the complex innovative nature of the subject. Consideration is also given to the fact that the study attempts to elicit tacit knowledge, perception, understanding and interpretation of a convoluted concept.

A qualitative research methodology was chosen as it permits a pragmatic and interpretative approach which is grounded on the perception and experience of people and focuses on the particular context of the research (Marshall & Rossman, 2006). However, the concept of open source component provision on a PaaS is a relatively novel one and as such, decisions had to be made on whether to employ deductive or inductive approach to examining the research concept. Since this area of research is new – in practice, little is known about the phenomenon of cloud computing and the concept of open source software component provision on a PaaS - the inductive approach was chosen as it permitted the researcher to examine a real aspect in the industry and identify a phenomenon which was then described using analytical tools such as theories, interviews and surveys. Empirical material based on the conceptual framework was collected, and then the framework was evaluated based on the empirical information. Confidence of the conceptual framework was enhanced based on the results of the empirical data.

The main goal of this research is to explore industry awareness and perceptions of PaaS, identify ideal users of the OSS PaaS offering by examining their basic characteristics such as the particular industry in which they operate, their company size, products/ services they offer to their customers and determine the types of services they desire. In addition to these, the effect of IPR is examined. Suitable literature and theories regarding the phenomenon have been explored in other to describe the main components of the study and explain the effects of intellectual property rights on OSS components offered on a PaaS and a conceptual framework has been created to abstract the theories. Considering the complexity of the concept, methodological triangulation (literature review, interview and survey) approach was used to investigate the research question in order to enhance confidence in the ensuing findings, mitigate the weaknesses of a single research method approach which is inherent in many qualitative studies and in so doing, validate the data through cross verification by using data from more than two source ((Webb, Campbell, Schwartz & Sechrest, 1966), (O'Donoghue & Punch, 2003)).

5.2 Selection of the research method and data collection

The selections of an appropriate research method hinges on several factors. Some key factors include: the nature of the phenomenon, the state of existing knowledge, and the types of questions to be asked ((Babbie, Survey Research Methods, 1973), (Babbie, 2008), (Dash, 2005)).

Specific characteristics of the research where taken into consideration when selecting the appropriate research method for executing the research project and answering the research questions. These characteristics include:

• Very limited empirical research has been done on cloud computing in general and the provision of OSS component on a PaaS in particular. Therefore there is very little existing literature on the subject.

• Platform as a service is a complex and novel endeavour. There is still a lot of ambiguity in the definition of cloud computing and its sub offerings. The idea of offering OSS components on a platform as a service combines the offering of a complex product as a service on a complex platform.

• The subject matter is very knowledge intensive and awareness of it is limited to a unique segment of society.

(Myers, 1997) and (Sala & Lynn, 2009) have proposed different methods for conducting qualitative and quantitative research. Examples include: Action research, grounded theory, Ethnography and case study research survey, history, archival analysis. The different research methods impose different techniques for collecting data such as: Interview, observation, archival research. Written data sources (includes both published and unpublished documents – articles, journals company reports, email messages, fax-). The various research methods answer different research questions and they have different control and time focus. One of the sub-questions in this study is explanatory since it involves *"How"*: ***How will license type affect the adoption and use of the service?*** To answer this question, both primary and secondary data have been employed. In primary data collection, the researcher collects data which is unpublished, unique to him/her and the research in question while secondary data concerns data that has already been collected and is available in various formats such as books, webcasts, periodicals, academic literature, journals, etc. In order to answer the research questions the appropriate data collection methods and sources were selected. Secondary data collection involved extensive perusal of relevant document data, archival analysis while a face to face interview was conducted to validate the information obtained from the secondary sources. The other two sub questions are descriptive questions as they have to do with *"What":* ***(a) what are the characteristics of the organizations which utilize such services? (b) What types of web application development services are most desirable?*** In other to answer these questions, a web survey was conducted on selected cloud computing forums and face to face interview was conducted in order to validate and improve the reliability of data collected from the survey.

5.2.1 Selection of interviewees

Three interviews were carried out; one with the founder and CEO of Tuxera Inc. who is also an expert in IPR law. The objective of this interview was to elicit expert opinion on the effect network on software IPR and if users of SaaS are bound by patent and copyright obligations.

The other two interviews were experts from software development companies. Both of these companies used and developed web based applications although one of them also offered a host of other solutions such as: device-independent internet-based services for media, entertainment and information delivery as well as social networking and ICT services on Business Process Management, Content and Case Management, eServices and Agile R&D.

The target companies were of different sizes, had different products, target market sizes and also different recourse and need for resources. For reasons of anonymity, the companies are labeled (A) and (B).

- Company (A) is a small company providing a platform for equity research analysis for investment banks. It has about fifteen employees with annual revenue of about 1 million Euros and its market is mainly Nordic European countries.
- Company (B) is much larger, and provides solutions for media and social network communities, mobile terminals and business solutions. It has a global operation with major interests in Scandinavia, China, and the USA. The company has about one thousand employees.

The objective of these interviews was to find out organizations perception of open source enabled cloud computing services and especially their preference for different platform services. Issues related to drivers and barriers to the platform service adoption were also raised.

5.3 Research design

The function of a research design is to ensure that the evidence obtained in data collection and analysis enables the researcher to answer the initial question as unambiguously as possible (Creswell, 2003). It is important because it provides a framework within which the research is conducted and enables both the researcher and subsequent readers of the research to be able to make sense of the study by understanding the role and relevance of the different components of the research.

However, obtaining relevant evidence requires that the researcher specifies the type of evidence needed to answer the research question and evaluate the concept or accurately describe the phenomenon. Failure to have a coherent research design early on in the study, may lead to unconvincing answers to the research question and inexact conclusions. The structure and logic of this study has been as follows:

- A research proposal and question was developed
- A literature review was conducted to establish understanding of the subject matter and address the question of how legal issues such as IPR affect the provision of OSS component on a PaaS.

- A theoretical framework was done in order to expound and ascertain the underlying theories driving the phenomenon.
- A conceptual framework was created to abstract the basic ideas put forward in the research proposal.
- Data collection techniques designed to answer the questions proposed in the research were selected and explained.
- This is followed by data analysis, interpretation and recommendations.

The research questions in this thesis have the form of “how” and “what” types of questions therefore, a document analysis method has been used to answer the “how” question while a cross-sectional study design type has been employed to answer the “what” type questions. For this reason, a survey was conducted across different cloud computing forums. The logic was to get respondents from different online forums which are all interested in cloud computing, but have different focus such as, the Azure cloud forum for windows cloud, google cloud computing group, Cloud Mobility, Sun Microsystems, Cloud Hosting & Service Providers Forum and Oracle cloud forum which focus on database in the clouds, cloud data centres and development platforms, Fedora forum, Cloud Computing Interoperability Forum (CCIF), Cloud Computing, Virtualization, Open Source, Web 3.0, Unified Computing, which give an open source and interoperability perspective on cloud computing services, Storage review.com which focuses on cloud storage.

The LinkedIn forum was used extensively in order to correspond with the different cloud computing groups and individual IT professionals. Finally, two semi structured face to face interviews were conducted with IT experts from two Finnish based software development and service companies in other to get the perspective from a typical software development company which would have to make a decision on whether or not to adopt such a platform and validate the survey result.

5.3.3 Data analysis

During this phase, the raw data is ordered and organized so that useful information can be gleaned from it. This process is important in enhancing an understanding of what the data does and does not contain and verifying the completeness and relevance of the data. Data analysis can be approached in a variety of ways. In order to make data analysis both rigorous and effective, this study systematically deployed certain strategies. Informal data analysis was performed during and immediately after each interview session. It included:

a. Writing down ideas and references to the underlying theory during the course of the interview.

b. A synopsis of the interview was done immediately after the interview was completed and initial findings were identified from each summary.

c. The recorded interview was transcribed and the transcribed document was compared with the audio record to ensure that the ideas expressed in the audio tape have been properly captured and represented. The texts were then carefully edited and forwarded to the interviewee for authenticity.

Verification of the information gathered was done by comparing information obtained from different sources. For example the information obtain from the interviewee on the effect of IPR issues on open source software distributed as a service via a network were compared with assertions made in other academic articles and in the FSF's rational document[39] , while data collected from the survey were compared to data from interviews.

The data was then grouped and sorted into relevant themes such as awareness, perception and motivation to engage in PaaS, the design and process model of a PaaS offering based on OSS and also the role open source software plays in the provision of PaaS. Organising the data into themes facilitates compacting and refining of the data so that interesting and relevant information can be gleaned in an objective and coherent fashion without the need to sort through all of the data. For each theme, all the narratives were compared extensively. Similarities and contrasts from the interviews, surveys and literature review are highlighted to either support or extend on underlying theories of the theme. The survey results are tallied, so that trends and perceptions are visible at a glance. These trends are highlighted in the write up of the data in the form of charts, graphs and textual representation in order to ensure that the patens and findings stand out in an objective manner.

5.4 The quality of the research

In order to ensure high quality of the data, three relevant tests in judging the quality of a research design have been considered: construct validity, content validity, and reliability. While it has not

[39] http://www.gnu.org/licenses/gpl3-final-rationale.pdf

been possible to apply all the theories which are relevant to the study of this phenomenon, every effort has been made to apply the most important theories and literature relevant to the study.

5.4.1 Construct validity

Construct validity aims at ensuring that there is a match between the theoretical concepts in the research and the specific metric or scale used in the research. The construct validity of an outcome measure is the extent to which that measure uses the theoretical attributes we seek to measure (Judd & Kenny, 1981). In other words, it refers to whether the scale or test measures the construct adequately. The scale seeks to translate the research ideas or questions into measurable elements by examining and measuring observable phenomena which reflect the underlying concept. Construct validity enables us to assess how well this has been accomplished. (Judd & Kenny, 1981), point out that;

> Every variable we measure is likely to reflect a variety of constructs as well as purely random errors. This is illustrated by the equation:
>
> $$\boldsymbol{Y = C1 + C_2 + ... C_R + E}$$
>
> *Y* represents some measured variable and the set of *C* refers to a set of unmeasured theoretical constructs that contribute to variations in *Y* while *E* refers to random errors in the measurement of *Y*

High construct validity requires that the measures adopted should truly assess the theories we desire to measure. The value of this construct should be substantial while the contributions of the other constructs which are not of cardinal interest to the research are minimal and the occurrence of random errors is kept at insignificant levels. Thus there are three necessary conditions for construct validity; *convergent validity, discriminant validity and reliability* ((Judd & Kenny, 1981), (Bagozzi;Yi;& Phillips, 1991)).

(O'Leary-Kelly & Vokurka, 1998) and (Yyn, 1994), pointed out that the major problem with construct validity is the subjective element involved in the interpretation of construct validity. The problem of research subjectivity has been dealt with in this research following two of the three methods proposed by (Yyn, 1994), namely (a) the use of multiple sources of evidence and (b) the establishment of a chain of evidence, such as clear links between the data gathered, the questions asked in both the interviews and survey and finally the conclusions drawn. Furthermore, the following steps have been taken to ensure discriminant validity, convergent validity and avoid research subjectivity; background theories and a questionnaire guide were employed when writing up the detailed documentation of the data and also to minimize errors and biases. A preliminary online survey was designed, verified, edited and validated by the research supervisors. The survey groups were then defined. Online cloud computing and open source forums where selected, and the researcher and supervisors gained membership in them. The researcher participated actively on the various topics of discussion in these forums in a bid to assess their level of maturity, areas of interests and level of expertise. The survey was then posted to the forums which were validated as relevant and credible. Continuous participation was necessary in order to assess the perception of the forum members, educe recommendations from them and make necessary changes to the questionnaire so as to elicit the appropriate information. As much as was possible, the concepts of this research were described to the survey respondents and interviewees before they were to fill out the survey or respond to the interview questions. In the case of the interview, a transcript of the interview was forwarded to the interviewee for confirmation. Finally, multiple data sources were used in this study in order to increase the reliability of the research by eliminating or minimizing the significance of random errors (noise) which may be inherent in a particular data source or collection method.

5.4.2 Content validity

Content Validity refers to the extent to which a measurement reflects the specific intended domain of content (Carmines & Zeller, 1991). In content validity, the elements of the test are assessed by trained individuals or subjects of the target population. The individuals make their judgments about the relevance of the items and the lucidity of their formulation. Their views are taken into consideration and adjustments are made to the tests in order to enable it address the problem in a more reliable manner. The content validity, like the face validity, is a qualitative measure of validity; it is normally not quantified with statistical methods. Content validity is concerned with a test’s ability to include or represent all of the content of a particular construct in this study, it was guaranteed by (a) having the research supervisors go through the research survey and make recommendations, (b) a sample of the survey was first released to some cloud forums and based on the responses, adjustments were made.

5.4.3 Reliability

According to (Judd & Kenny, 1981), the goal of reliability is to minimize errors and biases in a study. This view is also supported by (Yyn, 1994). In the equation; $Y = C1 + C_2 + \ldots C_R + E$, the occurrence of the random error E occurring is mitigated by the reliability test.

Reliability ensures that the same results will be arrived at should another researcher, employ the same procedure in conducting the same study. This is realized by using a standard set of questions which are clear and unambiguous and designed to elicit specific answers from the respondents. This is however not very easy in semi structured interviews which are subject to personal interpretation and are designed to elicit information from the interviewee which while being very useful; depend on the subject being interviewed and the particular circumstances of the interview. It was therefore important to select the interviewee carefully – those who possessed the most up to date information were preferred. For this study, professionals who were actively involved in software development and IPR law were chosen with the rationale being that, their opinions were most likely to reflect the current view of the whole industry. Test-Retest reliability method was employed in the online survey forms in other to verify that at two different points in time, the test result will stay consistent across similar groups of respondents. However, due to the limited time period (about 8 months) under which the research was conducted, the time interval between which the test was administered and re- administered was very short. In addition to the afore mentioned measures, the theoretical literature, online survey and web-references which were used throughout the research and in data collection are easily accessible thus ensuring repeatability and verifiability of the data collection procedure and result.

5.5 Limitations of the study

Completing a research project on a topic as complex as this study can be very complicated and time consuming. It also requires extensive use of resources which were not always within the means of the researcher. Although all efforts were made to ensure that the research result is of high quality, there where notwithstanding, some key factors which limited the execution and presentation of the research. These factors include:

- *Time:* the period over which the research was to be conducted was very restricted and on top of which the researcher had to juggle between the research work and other responsibilities.

- *Access to interviewees*: This presented one of the biggest challenges as it was never easy to get access to interview the most knowledgeable people in the industry. Other people who made themselves available for interview where often bounded by the need to keep trade secrets and comply with their company policies.
- In order to reach a global audience, it was necessary to use a web based survey. Unfortunately, while this method has a lot of advantages, it also limits the reliability of the responses as the researcher has no way of telling who responded to what and how many times a particular individual took the survey and therefore cannot authenticate the veracity of the responses.

5.6 Methods employed to mitigate the limitations of the study

The research study has employed different strategies to overcome the short comings of the chosen research method. Triangulation was identified as the most appropriate method for investigating the research problem as it offers the researcher the flexibility of using the strengths of both qualitative data collection techniques such as semi – structured interviews to elicit pertinent and in-depth information from the respondents and online survey which offers the researcher the possibility to collect a large amount of data across a wide geographically dispersed area within a short time and eliminates the need for the researcher to enter the data in a separate data base (a process which may be error prone). The data from web-based survey can also be automatically validated with the correct format imposed upon designing of the survey. The flexibility of the online survey (surveymonkey.com) also allow the researcher to make modifications to the survey such as options for the respondents to express other opinions not listed in the form, force the respondent to answer particular questions before moving to the next or before completing the survey. Web survey also makes it possible to target specific communities such as online open source communities and cloud computing forums.

The biggest problem with the online survey however is that of loss of control over the survey respondents. The quality of the data collected may be questionable as the identity of the respondents and therefore the quality of their answer is not known. In addition to this, respondents may make multiple entries into the survey and therefore influence the results towards their particular convictions. Techniques that were employed to mitigate some of the shortcomings of the research and data collection methods include:

- The use of a research framework and an interview guide to maintain the focus on data collection and reduce the amount of material to be processed.
- Secondly, preliminary analysis of the data was done as soon as it was collected in order to reduce the need for a huge amount of information.
- The survey was straight forward and broken down into sub sections which were designed to obtain specific responses. This also permitted the researcher to stay focused on the overall objective of the research.
- The problem of lack of control over the survey respondents was minimized by close and constant participation with some of the respondents, a continuous stream of email discussion was encouraged about the topic both within the forum and privately in the form of private email messages. A summary of the survey findings was made and submitted to the forums in order to obtain feedback. This helped the researcher to gauge the quality of answers the respondents were providing and also enabled the researcher to assess the level to which the respondents were able to understand the questions.
- Another control method used was the criteria in selecting the online forums in which the survey was distributed. It was ascertained that the main discussion in the forum was related to the general ideas of the survey, the forum had a sizeable population of active members and that the forum exercised adequate moderation policies.
- Finally, the same survey was presented to a select group of known respondents and their responses were compared to those from the forum. In order to mitigate the influence of bias or equivocal views on the findings and conclusions, the data interpretations were checked by supervisors and reviewed by peers who are also involved in similar research projects.

5.7 Summary

This chapter has explained how and why the research methods were selected, provided a description of their utilization within the research, and discussed research design issues. Next an in-depth design of data collection and analysis methods has been given and finally, a discussion of the quality of the research, the different tests which have been employed to ensure high quality data and a variety of measures which have been taken to mitigate the weaknesses of the research methods has been explained.

Chapter Six

6. Analysis, Implications and Conclusion

This chapter presents findings from an online survey and interviews which were geared towards finding out the characteristics of organisations which have or are likely to adopt a PaaS offering and their preference for the different types of cloud and platform services. It also examines the effect that the inclusion of OSS to the offer mix will have on their decision to adopt the service. The analysis was performed following the suggestions of the statistical service centre[40] and based on the conceptual model, diffusion of innovation theory, and relevant literature in the fields of cloud computing, open source software production, and information systems. The chapter presents the overall study by analysing and explaining its results and the implications.

Section 6.1 briefly explains the logic behind the selection of the different forums in which the survey was distributed and describes the activities and interests of these forums. 6.2 describes the composition of the survey respondents in terms of industry in which they work and size of their companies or organisations. 6.3 gives a measure of the level of awareness the respondents have about cloud computing in general, their perception about the future of cloud computing and the role that open source software will play in cloud based services. 6.4 examine the types of cloud and particularly platform services which have been adopted or are likely to be adopted by the respondents. The effect of including open source components in the PaaS offering is also discussed in this section while 6.5 discusses the perceived salutary and inhibitive elements of PaaS adoption. Results from the semi-structured interviews are used to corroborate findings or bring out greater insight in each section. Section 6.6 introduces the conclusions, while the major contributions of this study are described in section 6.7, and section 6.8 describes the implication of the findings in detail. Finally in section 6.9, discussions are presented which include the applications of the framework and benefits, and future research directions are suggested based on the findings of this research.

[40] http://www.reading.ac.uk/ssc/publications/guides/toppes.html

6.1 Description of respondent groups

A total of 15 different online forums were selected for the dissemination of the survey. Out of this number, 10 where primarily dedicated to cloud computing discussions while five were devoted to software development in general and open source software development. The forums had a fairly large membership - an average of two thousand members per group- however, only about 15% of the members were active although most of the discussions where very current. Not every forum member could be considered an expert in the general interest of the forum. Some members where actively working in the field of IT, while others where students, retired or just people interested in the subject.

The forums were chosen with the objective to obtain as many responses as possible from a relatively well informed group of respondents on the topic. Open source software and cloud computing are both novel and complicated concepts therefore it was important to address questions on this topic to forums where the members will comprehend the issues been discussed and have relevant views on the questions.

6.2 Description of survey respondents

Before examining how the respondents answered the survey, it is important to first of all examine their background. In this case, the industry in which they work and the size of the organization was examined in a bid to find out which group of people consider OSS component provision on a PaaS as being suitable for their adoption and also what motivates them to chose certain service options. This also permits the author to put the data obtained from the survey into perspective with reference to the background of the respondents. A list of different industry sectors where presented for the respondents to choose from.

Of the 57 people who responded to the survey, the majority (50%) where in the software development industry, while respondents who work in IT/Cloud services industry represented the next most populous group.

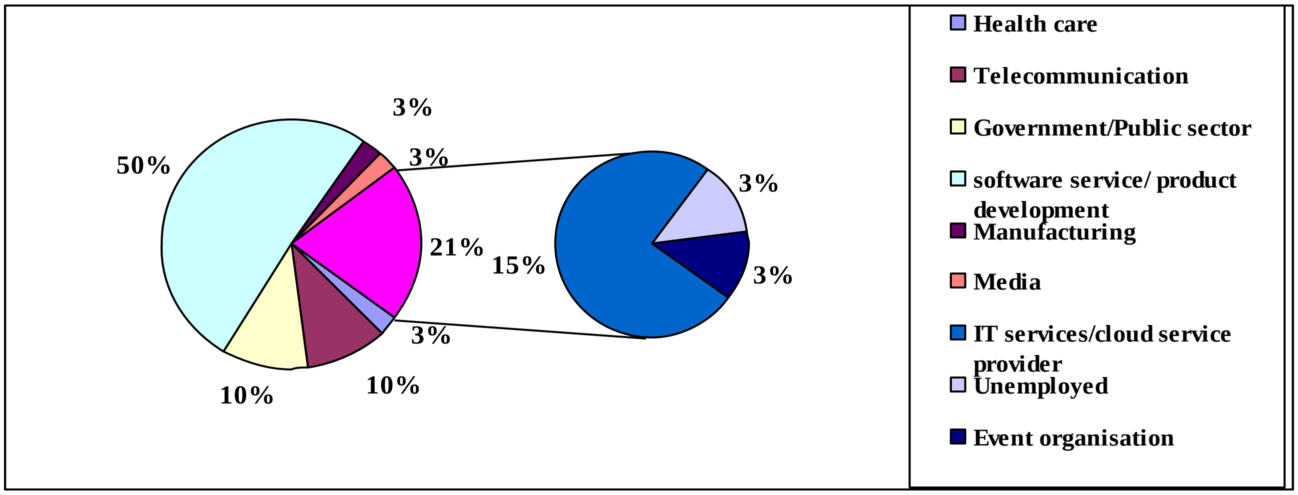


Figure 6.1 Distribution of respondents by industry

Figure 6.1 depicts a chart showing the distribution of the respondent by the industry in which they work. It can be seen that most of the respondents are from the software development industry while telecommunication institutions and government/public sector follow behind. This does not come as a surprise however, for several reasons; the nature of the topic in the first place is one which mostly relates to individuals and organisation involved with software development, regulation or high usage. In addition to this, one has to consider the environment within which the survey was disseminated. Majority of the participants in the different forums where people who were involved in software development or some related IT service provision; either as full time employees of established companies or freelance developers.

There was a huge variation in the sizes of the organisations within which the respondents worked. This proved to be a very interesting finding as the assumption before the research was that the concept of cloud computing and especially opens source enabled platform as a service will mostly appeal to small and medium sized companies while larger and more established companies will prefer to host all their infrastructure, develop and deploy from their own systems since they already have substantial sunk costs. This revelation required further investigation. It was important to find out exactly which services individual groups of respondents were interested in. Therefore, the different responses were filtered by size of the respondents company to find out if there was any correlation between the size of the company and choice of cloud or PaaS service.

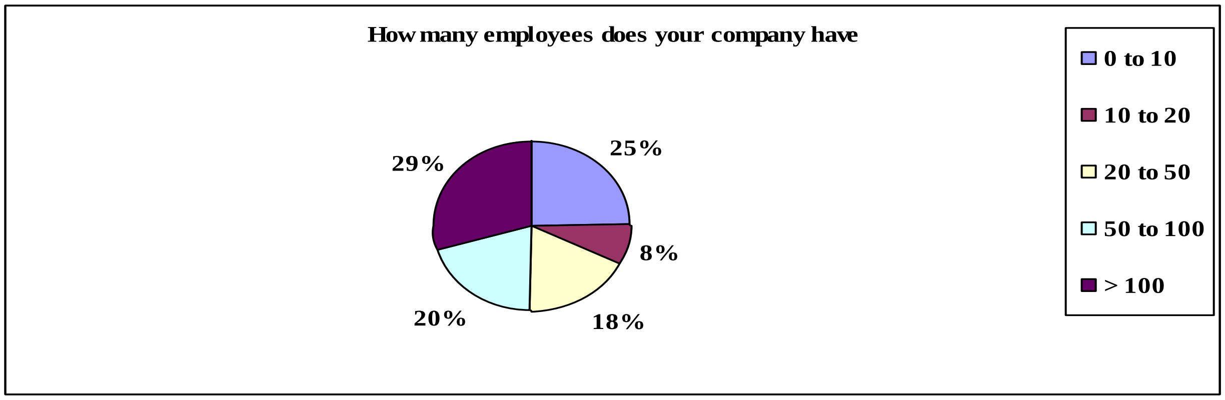


Figure 6.2 Representation of survey respondents by size of their

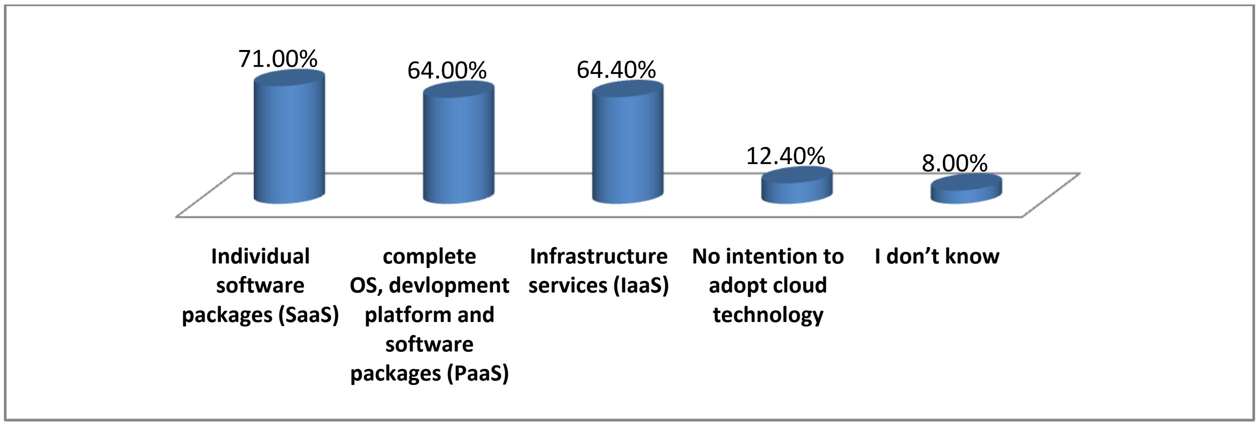


Figure 6.3 Respondents preference for different cloud computing services

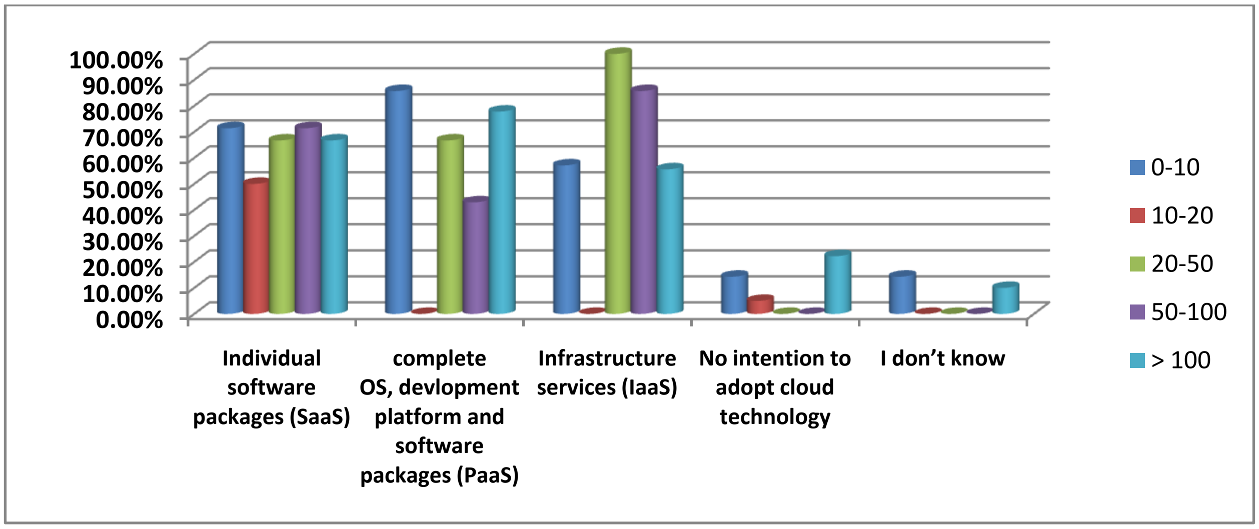


Figure 6.4 Preference for difference cloud computing service, discriminated by respondents' company sizes

Further examination revealed that, a significant percentage of the small and very large organisations have adopted or are likely to adopt all the three cloud service layers. However, both show a weaker desire to adopt *IaaS* layer (less than 60%) as opposed the *PaaS* and *SaaS* layers (more than 70%). The mid sized organisations all showed a very strong desire to adopt to adopt the *IaaS* layer with the exception of companies with between 10 and 20 employees. Interestingly, respondents from these companies only showed a desire to adopt *SaaS.* A cross tab between the size of the respondents organisations and their industry gives us greater insight as to the cloud service preference of organisations of various sizes, working in the various industries. In Figure 6.5 below, industries such as the media, cloud hosting and other IT service providers have not been included. There was no response from the media industry to this question, while individual examination of responses from cloud hosting and IT service provider organisations revealed that majority of the companies are midsized companies of between 20 to 100 employees.

The results from the survey broadly ties in with the response from the interviewees. Both interviewees found the PaaS to be a very appealing offer and attest that they were already using some form of SaaS. The interviewee from company (A) however stated that IaaS was very important for them at the moment since they did not own the infrastructure on which their solutions run and deployment, computing, web hosting and storage services have been outsourced to a third party company located in Helsinki.

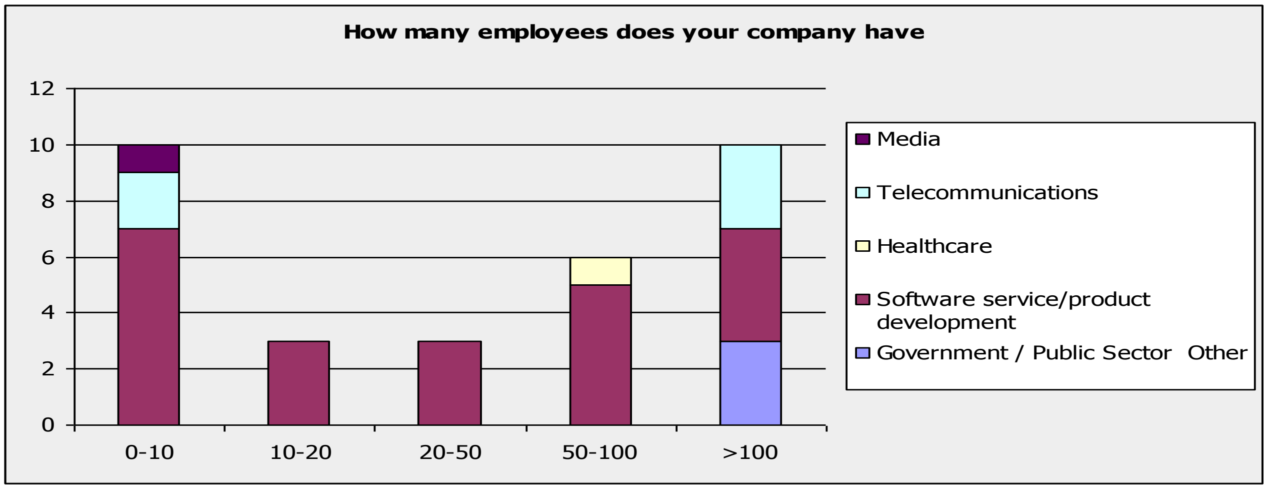


Figure 6.5 Size of respondents' organisation by industry

6.3 Awareness and perception of cloud computing and open source based cloud services

The survey result showed that while many respondents (95%) were generally aware of the concept of cloud computing and believe that its role and significance will increase in the next five years, (Figure 6.5 and 6.6) about 92% of them believe that the role and importance of open source software in cloud computing will increase over the same period. When asked which layer(s) of the Cloud the respondents have adopted or are most likely to adopt, only about 51% of the respondents answered that they are currently implementing or are planning to adopt some form of cloud service. With another 9% stating that they do not plan to adopt cloud services and about 40% not being able to choose from any of the listed cloud services. This figure is significant when we consider the forums in which the survey was distributed (most of the forum members were working in IT related service). The reason for this could be because, like most new concepts cloud computing is tickling the curiosity of many IT professionals in both the public and private sector and therefore driving them to seek for ways to benefit from the cloud by learning about its unique characteristics such as compatibility with existing systems and practices, relative advantage over alternative approaches and complexity of implementation (Rogers, 1995). Furthermore, prospective adopters may be allowing themselves time to be persuaded by both the technology, its suppliers (opinion leaders), and the experiences of earlier adopters ((Fichman, 1992), (Lazarsfeld & Katz, 1955), (Nahar, 2001)). Being pragmatic users of technology, this group of potential adopters are applying caution.

Both of the interviewees had some knowledge of the concepts of cloud computing and were very well grounded on the concept of open source software since it formed a major backbone to a majority of their offerings. The both saw future growth in significance of cloud computing and thought that open source software will play an important role in advancing cloud technology and constitute a core of the solutions offered in most of the cloud services. The representative of Company (A) though, stated that:

> For most start up companies and SMEs such as ours, cost is a significant factor. We will favour any technology which ensures that we carry out our operations more efficiently and at reduced cost. OSS has proved to be very stable and reliable and it does not cost us much therefore we always have a preference for it. If it is offered as an integral part of a platform solution, am sure my company will like to

adopt it provided the solution meets our needs, is of adequate quality and the price is reasonable.

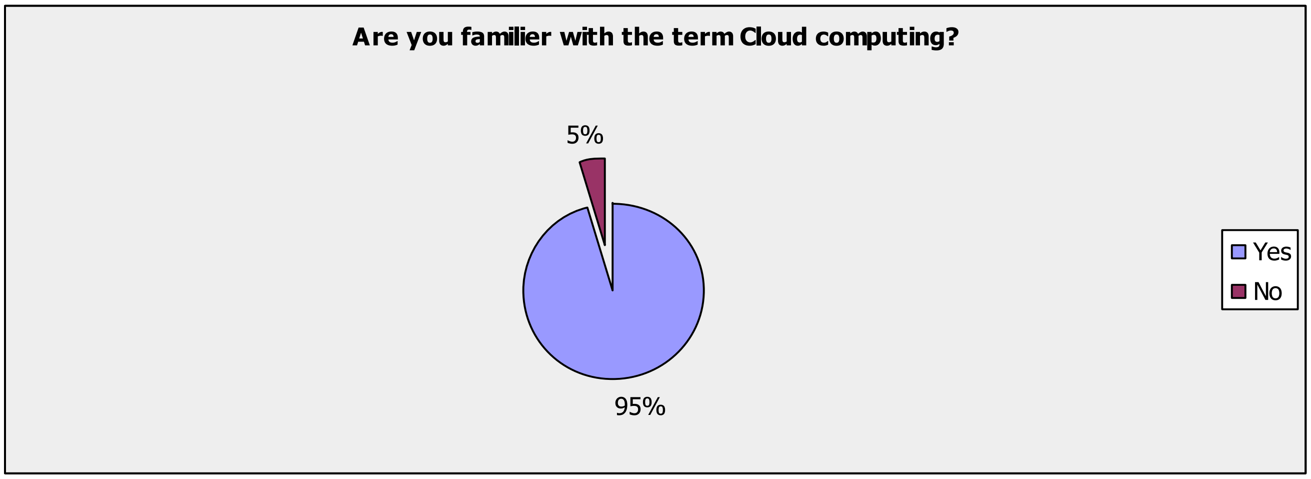


Figure 6.6 Level of awareness of cloud computing

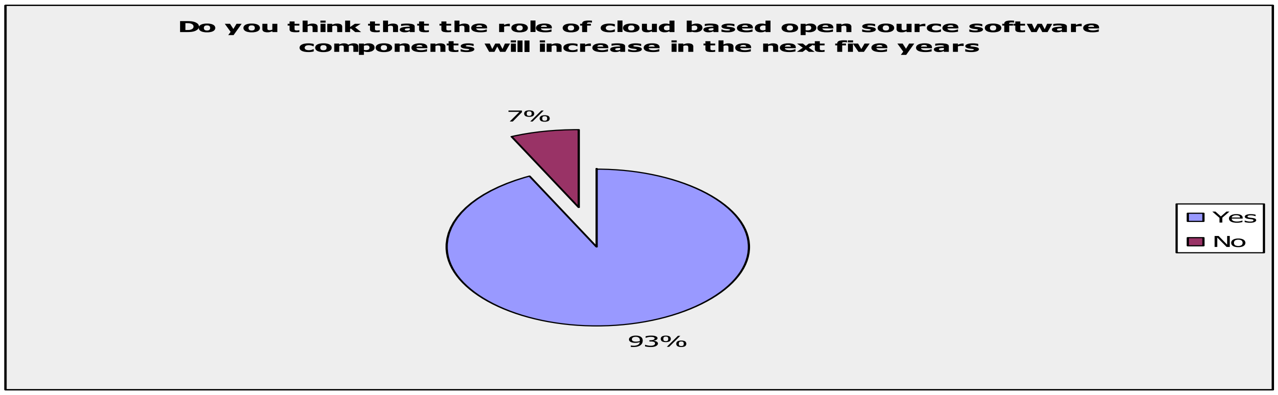


Figure 6.7 Respondents' perception on the increase in significance of OSS components in cloud services over the next five years

6.4 Adoption of cloud platform service and the effect of including open source components on PaaS

Since the main focus of the study is on open source software and cloud platform as a service, it was necessary to further investigate respondents' adoption of different PaaS services. This enables us to decipher what particular cloud service is considered important among the different groups of respondents.

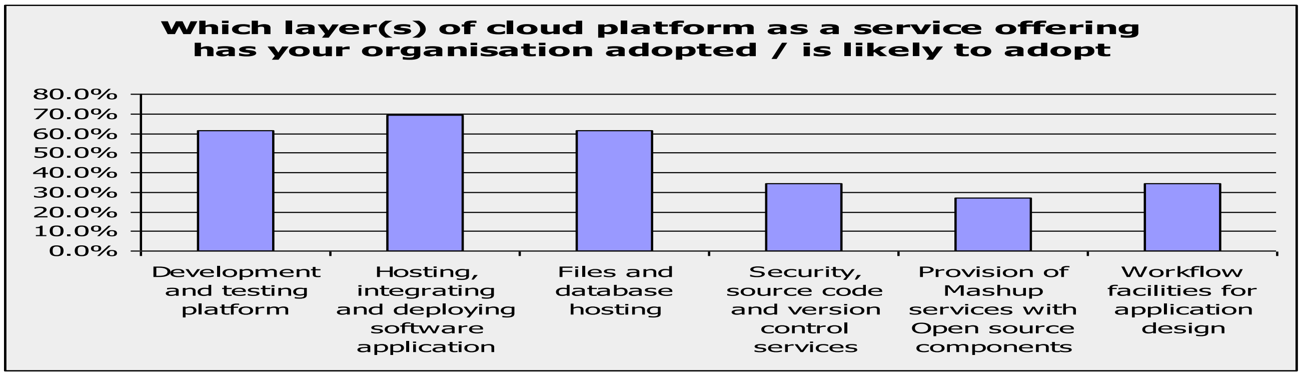


Figure 6.8 Preference for different cloud platform services

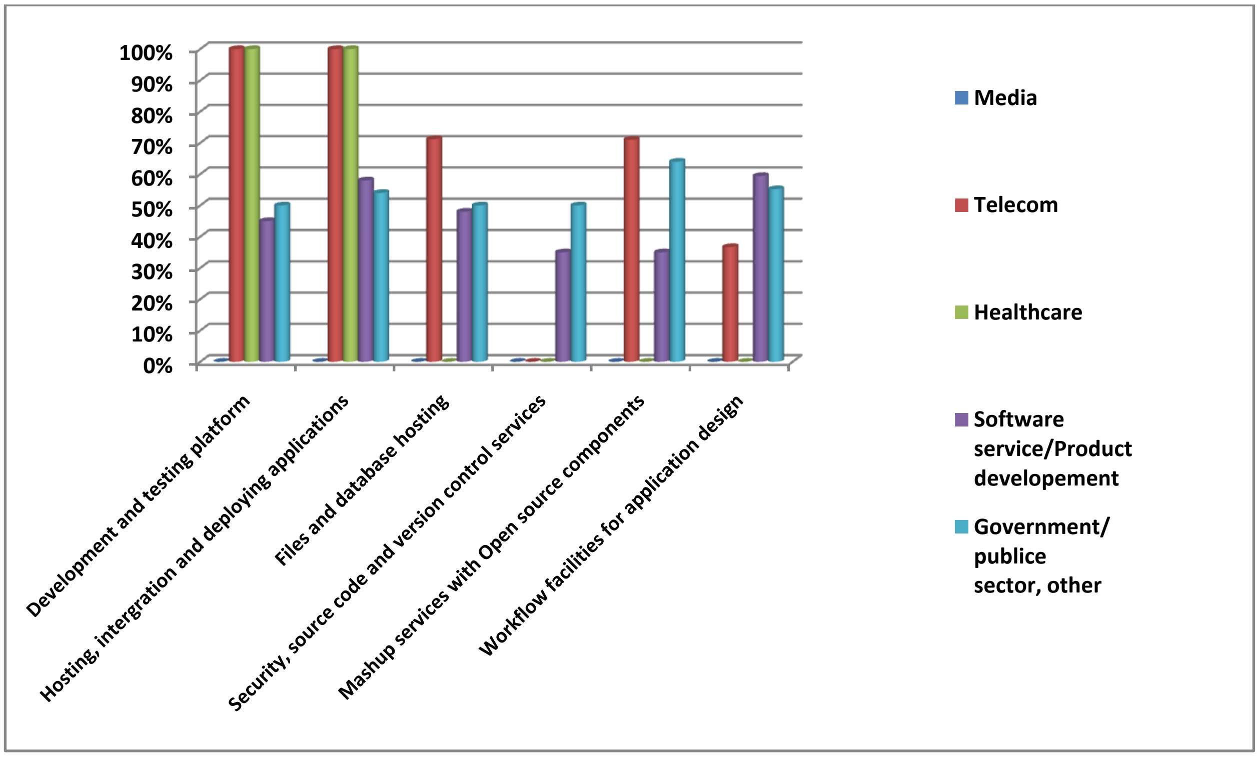


Figure 6.9 Preference for different cloud platform services, discriminated by size of respondents' company

From the survey, cloud hosting, integration and application deployment services where considered the most popular option by the respondents while development and testing platform and files and database hosting followed next. Security, source code and version control services proved to be the least sought for services but this could be due to the fact that most often, these services are usually bundled and offered with development and testing platform.

Company (A) representative thought that an integrated development platform is very important especially if it also have functionalities for version control, collaboration tools and web content management. He stated that at the moment his company is using Joomla for content management and they are paying another company for collaboration services. He however pointed out that software testing is considered a very important aspect and they generally wish to have greater control over how this is done therefore if they have to do their testing on the platform, it should be such that they can customise it to suit their needs.

Company B representative on the other hand sited security, source code and version control as being too important to outsource to a third party platform.

By analysing the respondents' preference for different PaaS services based on their organisation, we realised that respondents from the public sector and software service/ product development industry both selected all the different platform services. However, the desire to adopt security, source code and version control services and mash up services with open source components was lowest for software development companies. This again could be as a result of the fact that these services are normally considered to be part of development and testing platform services. The telecom and healthcare industry showed one hundred percent adoption of development and testing platform and hosting, integration and deployment services. However, these figures may not be very significant if we consider that there was only one respondent from the healthcare industry and four from the telecom industry. No respondent from the media industry selected a PaaS offering.

When asked how the inclusion of open source software components on the PaaS offering will affect the respondents adoption decision making process, 47% of the respondents answered that it would have a positive effect while about 34% said it would not have any effect and a further 19% said it would have a strong positive effect. This result was strongly corroborated during the interviews were both respondents welcomed the idea. This wide acceptability of open source software in an offering mix presents a shift in users' willingness to adopt open source software. This is not explained only by the efficiencies of OSS ((Raymond, 2001), (Krishnamurthy, 2003)) but also by the shift in responsibility to manage and maintain software support from the users to the platform provider coupled with the opportunity to avoid some of the restrictions associated with OSS licences through the network loophole[41]. These rationales make the idea of adopting open source software as part of a platform service offering much more appealing and it validates and explains the response to the previous question on the how significant open source software will be in cloud services over the next five years where most respondents (93%) were of the opinion that there will a significant increase in the use of open source software in cloud services.

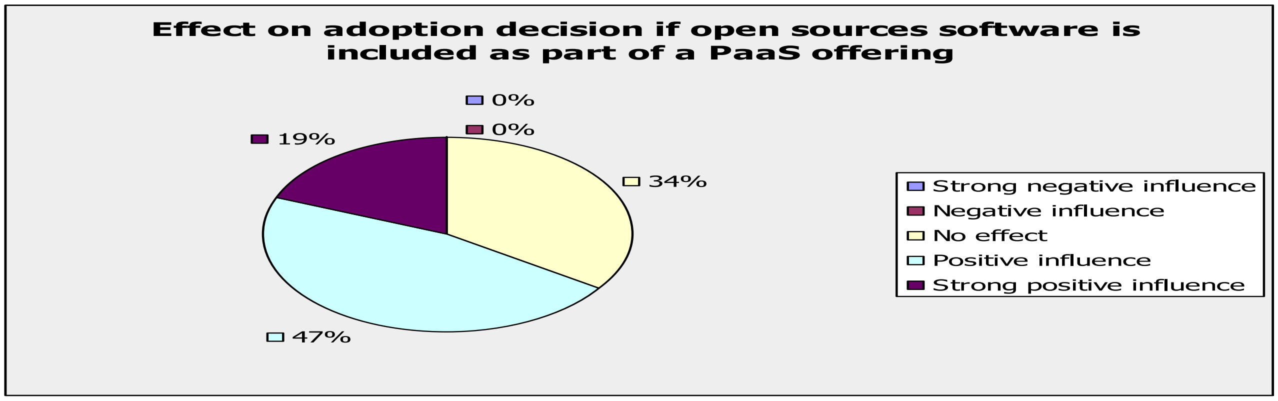


Figure 6.10 Influence of OSS on decision to adopt PaaS

6.5 Drivers and barriers to PaaS adoption

6.5.1 Drivers of PaaS adoption

In order to find out what aspects adopters or will be adopters perceives as beneficial and hence drivers to the adoption of platform as a service, a list of beneficial elements associated with PaaS was suggested to the respondents and they were asked to rate whether they agree, disagree or did not know the answer. When asked if the respondents believe that PaaS leads to a reduction in upfront capital expense, 75% percent of the respondents agreed while 11% disagreed with the remaining

[41] www.gnu.org/licence/gpl3-final-rationale.pdf
GNU has no networked future http://www.linux-mag.com/id/3017

[41] Tim O'reilly claims here that the decision not to close the SaaS loop hole could not be made in GPL v3 because many SaaS providers had already incorporated OSS in their offerings.
http://radar.oreilly.com/archives/2007/07/the-gpl-and-sof-1.html

14% of the respondents either stating that they did not know the answer or skipped the question. (Armbrust, et al., 2009), concur that lowered financial barrier to entry is one of the major reasons why cloud computing services are being considered popular this days especially for small and medium size enterprises. However, they claim that although the economic appeal of cloud computing is often described as converting Capex to Opex, it is the concept of *pay as you go or usage based pricing* which directly captures the economic benefit to the user. The usage based pricing model mitigates the problems of both over provisioning and underutilisation which may lead to redundant costs and loss of opportunities ((Weis, 2007), (Armbrust, ym., 2009)).

Elasticity of cloud services gives developers the opportunity to deploy and operate innovative and untried services without being overly concerned about over provisioning for services whose popularity may not meet their prediction and thus wasting resources, or under provisioning and thus loosing valuable opportunities. Instead, developers have the flexibility to scale their services up or down as the situation demands ((Tucker, 2010), (Armbrust, ym., 2009), (Smith, 2010)). Furthermore, cloud architectural flexibility enables vendors to offer their customers a range of deployment options, including hosting for the services they require, and allows users to choose from a range of prebuilt features or choose which features of the application they will implement themselves. This can reduce the architectural liabilities for end users who are developing services. This view was reflected in the survey with 79% of the respondents agreeing that PaaS Increases flexibility of the adopter company while 7% did not agree with this assertion and 14% could not agree or disagree with the assertion.

The outcome of this question is hardly surprising and it is emphasised in the responses from the next question where the respondents were required to state whether they agree, disagree or did not know the answer to the assertion that PaaS allows the company to focus on its core business. About 75% of the respondents agreed to this statement with 11% of the respondents disagreeing and 14% stating that they did not know whether to agree with it or not.

Cloud computing enables software developers to improve their operational efficiency and access business applications that were previously only available to large enterprises by permitting them to outsource part of their operations to a development platform provider for a fee, and access cloud based applications (Banks;Erickson;& Rhodes, 2009). By taking away routine and secondary tasks such as infrastructure and system maintenance, managing upgrades and patches, ensuring system availability, application and data security, the platform provider permits the developer to focus on their core operations. In addition to these, some PaaS vendors also provide prebuilt business functionality that enable users to jump-start projects by avoiding to build everything from scratch[42]. This releases capital and human resources to the user which can be employed in further innovative projects and thus make them more flexible in an ever changing industry.

Solvito[43] is an example of a company which has obtained greater flexibility and the opportunity to place greater emphasis on their core business - developing and marketing their CRM solution *Ufficio* – by adopting a development and hosting platform since neither the operation nor the management of the IT infrastructure requires any resources. Furthermore, Platform-as-a-Service helps Solvito pave the way for new, more flexible marketing options and also brings further added value such as a quick rollout and the possibility to offer demo versions of *Ufficio* to interested customers in a very short time. The downside to this assertion is the problem of loss of technical skill as a result of outsourcing.

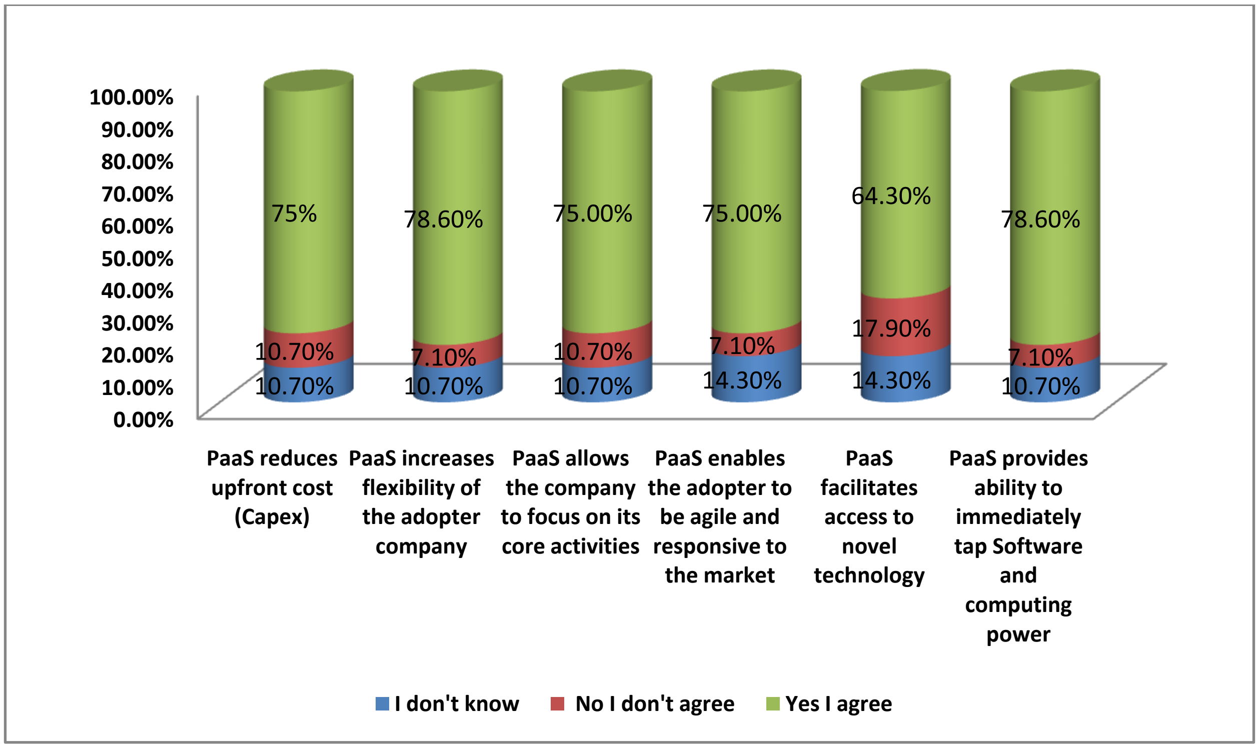


Figure 6.11 Drivers of PaaS adoption

[42] http://www.salesforce.com/paas/benefits-of-paas/
[43] http://www.grouplive.com/en/solutions-for-isvs.html: Solvito is a customer of GROUP Live which is a platform as a service provider.

75% of the respondents supported the assertion that PaaS enables adopters to be agile and responsive to the market while a further 7% could not agree with this statement and another 18% indicated that they did not know if it is true or not. PaaS vendors often offer online communities where developers can get ideas, share best practices, and seek advice from others. This approach is very similar to a network of open source developers but retains the advantage that developers are under no obligation to release their source code. The direction of the industry can be gleaned from participation with a vibrant community which may serve as both a source of innovation, market for the developers' product and reference to potential market. The prospect of the developers being able to carry out testing simultaneously with development, gauge users' acceptance by virtue of immediate access to a large pool of potential users and make changes on the fly offers platform adopters a level of agility which traditional software developers may lack. As the survey indicated, a significant percentage of the respondents agreed with this assertion. However the down side of this assertion is the restrictions placed on the adopter by the limitation of the PaaS provider. Where the provider's community is relatively small, the likelihood of reaching a mass market is curtailed and quality and quantity of information being shared may be affected. In a case were the users' activities are strongly tied to the provider, the performance of the provider in the wider market directly impacts the platform user.

63% of the respondents concurred with the suggestion that PaaS facilitates access to new technologies while another 19% either skipped the question or stated that they did not know if this is true or not and 18% could not agree with the assertion. A cloud development platform provides an integrated, end to end development environment where developers can build, test deploy and host web applications ((Motahari-Nezhad, Stephenson & Singhal, 2009), (Grossman & Gu, 2009)). By leveraging the multi-tenancy paradigm of the platform, the PaaS provider is able to acquire and offer expensive and innovated technology to users who may otherwise not be able to afford it. The PaaS provider is also able to stay at the cutting edge of technological innovation by creating and collaborating with large ecosystems of developers and open source communities. An example of an organisation which has benefited from a PaaS provider by being able to tap into new and superior technology is Solvito[44]. Solvito accesses new technology from the GROUP Live platform and the IBM infrastructure and used it to successfully convert the Lotus Notes-based CRM application *Ufficio* into a cloud-capable solution within a very short time and access a large SaaS market. Platform providers offer an opportunity for companies like solvito to access and use new technology

[44] http://www.grouplive.com/en/solutions-for-isvs.html

without the need to incur the huge costs which are involved, or significantly improve their knowhow. However their ability to access new technologies could be restricted by the limitations of their provider.

About 79% of the respondents believe that PaaS provides the ability to immediately tap into software and computing power while 14 % responded that they did not know if this assertion is true or not. Sales force.com claim in their website that one of the biggest advantages of adopting cloud computing is its ability to scale up or down as the needs of the adopter fluctuates[45]. This assertion was largely confirmed by (Armbrust, ym., 2009). PaaS also enables users to harness computing power by ensuring global availability (Banks, Erickson & Rhodes, 2009). According to a report from the computing and communication industry association (CCIA)[46], cloud computing ensures that work projects can “follow the sun”: this is achieved by storing resources remotely so that companies can accomplish greater productivity and efficiency by allowing users and employees around the world to collaborate on projects on top of a single platform. Conversely, workloads can “follow the moon”: the actual storage and computation of data is migrated to servers and data centres in locations that require the least energy usage.

The result of the survey generally tied in with the opinions of the interviewees. However the representative of company (A) thought that Cost was probably the most crucial factor. He stated that:

> Cost is an optimisation issue which could be solved. It should be looked into carefully. If the premium is too high, then it may be better to do everything in house.

He however cited the gains made in being able to tap on to privileged technologies and ability for the organisation to outsource routine services and focus on their core businesses.

> Software companies should outsource operational routine which consume a lot of time in small firms especially if you have to deploy the service. For example SaaS providers can outsource version control, collaboration and content management services and focus on development.

[45] http://www.salesforce.com/paas/benefits-of-paas/
[46] http://www.ccianet.org/CCIA/files/ccLibraryFiles/Filename/000000000151/Cloud_Computing.pdf

The representative of company (B) also concurred that cost is an important driver. In addition to this, he also cited the fact that PaaS permits companies to focus on their core activities, be more agile and responsive to the market as important driving factors.

6.5.2 Barriers to PaaS adoption

Information concerning the elements that hinder the adoption of platform as a service were obtained by means of the online survey and supported by semi structured interviews. Ten different inhibiting factors were identified and listed with the survey respondents asked to select how important this element was as a deterrent to the adoption decision making process. The options available for the respondent to choose from included: Very important, important, slightly important and not important with "*very important*" signifying that the respondent find this elements to be of real and significant concern to them, while "*not important*" signifies that the factor does not present any real threat or there are adequate measures to moderate the threats they pose or re-enforce the weaknesses they may possess.

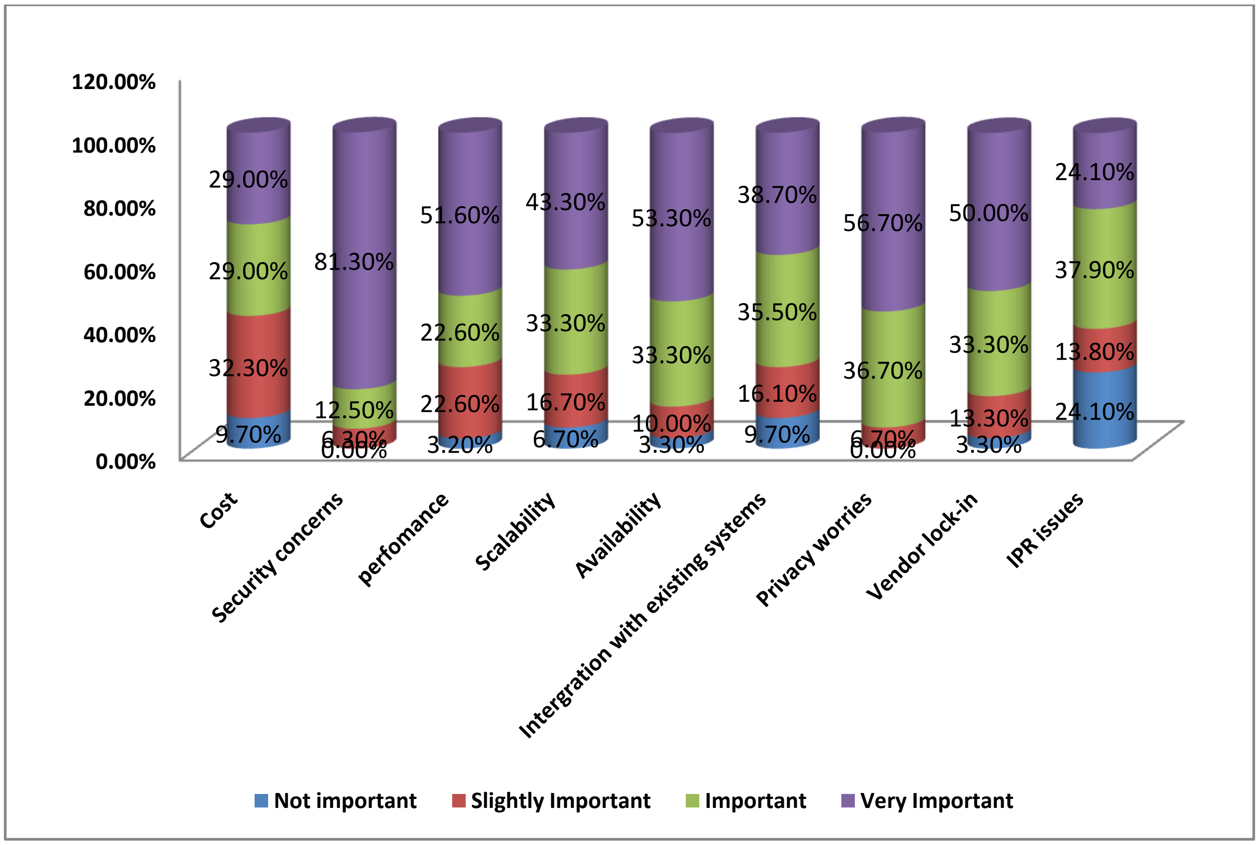


Figure 6.12 Barriers to PaaS adoption

Among the inhibiting factors which were presented to the respondents, security concerns, privacy issues, availability of service, vendor lock in, performance and loss of control over service/ data. privacy issues where considered to be most important with over 50% of the respondents in each case, stating that they considered these factors to be very important in their decision making process. Security proved to be the biggest problem for adopters with about 81% of the respondents citing it as a major problem and only 6% being able to consider it as slightly important. None of respondents considered it unimportant.

Security has always been a major issue in open system architecture it has also been cited by other researchers and authors such as ((Smith, 2010), (Armbrust, ym., 2009)) Gartner report of June 2008 entitled: *Assessing the security risks of cloud computing* [47] enumerated seven security issues regarding cloud computing. The specific issues include;

- Users' loss control over how to restrict access privilege to their data in the cloud and how the data is being managed: this is also presented as a separate inhibiting factor in the survey and unsurprisingly, majority of the respondents stated that it was also a very important inhibiting factor. Potential adopters are generally uncomfortable with the idea of having their critical information and application in the custody of a third party.
- Data segregation in multi tenant architecture: Privacy in cloud architecture is a challenge. According to (Brodkin, 2008), while encryption technology is effective in ensuring data privacy in the cloud, it is nearly impossible to guarantee 100% security and privacy protection against all possible sources of violation, including the inevitable software bugs, the growing sophistication of the hackers, inadequate procedures and human errors.
- Long-term viability of the cloud platform provider is another source for concern to adopters while other issues such as difficulties in investigating inappropriate or illegal activities in cloud, risks of data loss and unavailability of service due to disaster and illegal human activities and inefficient disaster recovery plans constitutes another source of anxiety to users.
- Problems related to compliance with regulations, especially those concerned with the locations of stored data and adherence to privacy regulations is another key security problem according to the Gartner report[48]. (Kim, 2009), pointed out that enterprise users must

[47] http://www.computerweekly.com/Articles/2009/02/24/234988/google-mail-collapses.htm
[48] http://www.gartner.com/DisplayDocument?id=685308

> maintain business legal documents and assure their integrity in order to comply with various laws, such as Sarbanes-Oxley (SOX) and health insurance portability and accounting act (HIPAA). While cloud computing vendors have to adopt technologies to ensure that their enterprise users' data satisfy their compliance requirements.

The issue of security was also mentioned by both interviewees although it was more important for the representative of company (B) than for company (A) representative. Company B representative stated that:

> Security is always something to worry about and it is not just technical security and privacy but also policies and standards.

Availability: Amazon S3, Google Gmail, Citrix's GoToMeeting and RIM's BlackBerry services have all suffered outages of from two to eighteen hours within the past two years. While these were not exactly catastrophic incidences, many researchers and critics of cloud computing have been quick to point out the potential disasters which such loss of service availability may cause (Hoover & Martin, 2008).

The lack of a set of universal standards and interfaces among cloud service providers increase the risk of vendor lock-in. (Armbrust, ym., 2009) asserts that this is an important deterrent for potential platform adopters. This claim is validated by the survey with 50% of the survey respondents citing this as a very important factor, and about 33% citing it as important. However, the creation of global cloud computing standards and audit groups such as the Open Cloud Standards Incubator (OCSI), the cloud computing standards forum and the Enterprise Cloud Buyers Council (ECBC) will make this problem less significant for future adopters.

Both interviewees thought that issues such as availability, scalability and performance were not very crucial as most of the problems associated with them have already been solved. The representative from company A stated that their clients were not overtly worried about availability and they have about 99% uptime. They use about 3 or 4 virtual servers in order to mitigate the risk of service outage. And warn their clients in the event that such occurrence cannot be avoided.

Service cost and Intellectual property right issues were thought to be relatively unimportant in the PaaS concept. Less than 30% of the respondents thought that this was a very important problem.

And up to 24% of the respondents thought that IPR issues were not important in cloud platform service provision. However, 38% of the respondents still believed that IPR issues were important. (Armbrust, ym., 2009), pointed out that the fine grain economic models enabled by cloud computing makes trade off decisions more fluid and serves to transfer investment risks to the cloud service vendor. However cost was not dismissed as a trivial element by the representative of company A. Rather he claimed that it may be the major deciding element on whether the company adopts a platform providers services or not.

6.6 Conclusions

The purpose of this thesis was to find out how open source software can be offered as part of a service mix for web application developers. Three sub questions were generated from the main question:

a) the characteristics of organizations which are apt to adopt and utilize open source software components provided on a platform as a service services, b) the type of platform services which are most desirable and c) analyze how license types and IPR affect the adoption and use of the service(s).

To answer the above research question, in depth literature review was conducted on open source software development methods, business models and especially licences. Furthermore, literature on the different elements of cloud computing with an emphasis on platform as a service was conducted in order to lay the foundation for the research concept. A conceptual framework was proposed to abstract the concept and identify the key elements of the theory. Finally an online survey combined with semi structured interviews has been conducted in other to investigate individual awareness and perceptions of platform as a service, find out the characteristics of the organisations that adopt such platforms, the elements which drive them to make the adoption decision, the factors which act as deterrents to adoption and also the specific types of platform services which are desired.

6.7 Major contributions

This research contributes to the understanding of open source component provision on a software development platform, it also brings to light, the issue of open source patents and copyright licences in a network environment. It attempts to conceptualise a business model for providing platform as a service with open source software components. The framework reveals the relevant aspects of the concept and elaborates on the interrelationships between the different elements of the business model. The process design is examined and key actors and their roles are identified. This is an important new topic in open source software and cloud computing research. The phases, requisite elements, actors, the performance measures, challenges and the expected outcomes identified in this study are necessary for a company that is intending to harness the power of open source software and offer application development service platforms as a business model.

This study made investigations on the diffusion of innovation theory and presented ways in which it supports the conception, diffusion adoption and implementation of innovative technologies. This investigation is particularly important for the open source platform as a service business model because, it helps us to understand how the consumer and supplier can learn from each other, the types of services which can be offered, how they may be offered, their limitations, how the model as a whole can be managed and the potential of the market. This research began with developing an integrated perspective on many issues that fall under both open source software and the cloud computing umbrella. The study initially expounds on these concepts as relatively separate ideas in order to fully explicate their characteristics. Later both concepts are put together, by highlighting their mutual strengths and complementary nature which permits circumvention of their separate weaknesses and arrived at a powerful synergy which forms the idea of the business model.

Through an in-depth investigation and analysis, this study contributes in terms of both theory and practice. At the theoretical level, it indicates the existing knowledge in the following ways:

- It contributes to the scientific understanding of open source software and community evolution and development; it explains the intricacies of legal issues surrounding the development, use and distribution of open source software.
- It contributes to the scientific understanding of the essential cloud computing characteristics and service models.
- It delivers a conceptual framework of open source software enabled platform as a service business model which is composed of three distinct layers: The external actors (suppliers and partners, customers), core business activities and the peripheral support systems. This framework can be used as a basis for further research.

- It develops a broad view of providing open source software components on a platform as service model. Rather than focusing on a single aspect of the concept in this framework, the main activities related to the different actors, services and market segment, were identified.

On a practical level, the study contributes to the existing knowledge in the following ways:

- It provides a detailed and practical blueprint of the piece which constitutes an open source enabled platform as a service business model, highlighting the roles, relationships and relevance of the different elements of the model.
- It identifies the different platform services, some of the software functionalities which are provided on a platform and current open source software components which are being used to provide prebuilt business functionalities.
- It highlights the preference of different cloud computing services by companies operating in different areas of the industry and also attempt to discriminate companies preferred service option based on their size.
- It also identifies the major challenges related to cloud computing services: security, availability, interoperability and privacy.

This study is one of the few studies which attempts to bring together the benefits of recouping the advantages of a community based software production paradigm and an integrated software development environment which releases the innovative potential of developers by relieving them of the burden peripheral but necessary activities of the development process.

6.8 Implications

This study proposes an approach on how to effectively leverage the advantages of open source software while mitigating some of its draw backs and targeting a more sophisticated user segment. Increased competition, a need to produce user centric application at competitive costs, a strong desire to reduce development cycles, tap unto a wide pool of human resource and reach a global market makes it necessary to investigate the idea of leveraging platform as a service and open source software to create new business models.

The research further reveals the major deterrents to cloud service adoption in general and platform as a service in particular. Among the key inhibitive elements are:

- Vendor lock- in: An online survey revealed that about 50% of the respondents were wary of being tied to a particular provider's platform because of difficulties to migrate applications from one platform to another. This is a major problem which cloud providers are trying to overcome by introducing open cloud standards and encouraging interoperability policies.
- Security, availability, performance, loss of control and privacy: these were identified as the most significant areas of concern. Users want to have control and be sure about the safety of their data and applications on the web. In addition to this, compliance with national and regional data privacy policies often means that users and service providers have to follow specific regulations when using and storing customer data. Virus threat is also another security threat on the internet. Some measures have been developed to mitigate the risks involved such as CloudAV which is an in-cloud antivirus system that includes a lightweight, cross-platform host agent (Win32, Linux, FreeBSD, Sendmail/Postfix milter, Nokia Maemo) and a network service with ten antivirus engines (Avast, AVG, BitDefender, ClamAV, F-Prot, F-Secure, Kaspersky, McAfee, Symantec, and Trend Micro) and two behavioural detection engines (Norman Sandbox, CWSandbox) (Oberheide et al 2008). Specifications in the service level agreement usually informs the user on how their data is going to be stored and the availability and security measures which are taken to safeguard their information.
- Patents and Copyright issues have not been completely discarded as could be seen in figure 6.12. However, the responsibility for ensuring compliance lies more with the service provider.

The research reveals that although development on cloud computing is in its nascent stage, there is a lot of prospect for the future and avenues for growth in the cloud. The significance of cloud computing technology and the role of open source software in the cloud are bound to increase in the near future (see figure 6.7). Companies which harness the technology and build competitive business models around it may emerge as new leaders in the software industry. It is important for companies to gain an understanding of the both open source software concept and cloud computing in order to identify ways of leveraging these ideas to create new business opportunities. The results of this research are useful in providing such understanding.

6.9 Discussions

6.9.1 Applications of the framework and benefits

The open source enabled platform as a service framework is a generic business model framework and may not be completely applicable to every business case. It can nonetheless be adapted to fit he needs and the specific situations of companies that wish to use it. Utilising open source software as a service reduces the cost and development life cycle of software. It also puts vital tools for collaboration and software development at the disposal of developers in a manner that is convenient for them.

Furthermore, adopting an ecosystem approach for software development and provisioning creates a synergistic effect that may lead to production efficiency. Technology alone does not play a significant role in all aspects of the model. While it is true that having a sound technological concept combined with a strong business plan goes some way in ensuring success, other important variables must also be considered such as relationships with partners and customers. In this model, a lot of value can be harnessed by participating profoundly with open source communities and platform user forums. For example Rackspace[49] is a cloud service provider that has built a user community by creating a forum where users can communicate on issues such as application integration and performance tweaks. This allows the establishment of trust between customers and the service provider. It also provides an informal support channel, a beta testing community for both the platform adopters and the platform service providers and as an avenue for innovative ideas.

6.9.2 Future research directions

This research delivers a business model framework for offering open source components as a part of a PaaS offering. The ideas put forward in this research can be used as a basis for further research in the field of open source software or cloud computing. Further theoretical and empirical research will be helpful in refining the framework. The framework discussed in this study is a business model framework. However in order to optimize the practical value of the study, further work needs to be done in order to develop a business process model for the concept with in-depth case study analysis carried out on existing business operating on similar concepts.

[49] http://www.rackspace.com/index.php

It is also necessary to carry out more detail research on specific open source software services which can be provision using cloud platforms and the open source software components which are used to enable such services.

Another interesting research which may stem from this study is an investigation into the different billing strategies for platform services and adopters perspectives on them.

As explained in chapter three of this study one of the limitations of the classical innovation diffusion theory is the assumption that individuals are adopting innovations for their independent use. It ignores the effect of user interdependencies in the adoption decision making process. (Lazarsfeld & Katz, 1955) illustrated the effect of network externalities in innovation diffusion. (Fichman, 1992), pointed out that this is even more evident in information technology innovations where the value of use to any single adopter is a function of the size of the network of users. (Tsikriktsis, Lanzolla & Frohlich, 2004) emphasised on the bandwagon effect and the use of mass media as an effective information channel to build a critical mass of innovation adopters. However while this theories affirm that the nature of a technological innovation and the effect of a chosen communication channel play a vital role in determining the readiness with which individuals and organisations take up new technologies, it does not explain failures in perpetual adoption and implementation. Therefore, it is important to investigate the factors that drive continuous adoption and implementation of open source enabled platform as a service.

Bibliography

Alt, R., & Zimmermann, H.-D. (2001, January 1). Introduction to Special Section - Business Models. *Electronic Markets* , 3 - 9.

Andreas, G., Timothy, J., & Bobby, L. (2006). *The Economics of Information: Google Apps ForEnterprise Installed Solution.* Cambridge, MA, USA: MIT Sloan.

Andrew, V. d., Harold, A., & Poole, M. S. (2000). *Research on the Management of Innovation.* New York: Oxford University Press.

Armbrust, M., Fox, A., Griffith, R., Joseph, A., Katz, R., Konwinski, A., et al. (2009, March 2). *Above the Clouds: A Berkeley View of Cloud Computing.* Retrieved April 11, 2010, from System News : http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.pdf

Attewell, P. (1999). Technology Diffusion and Organizational Learning: The Case of Business Computing. *Organization Science* , 1-19.

Axelrod, R. (1984). *The Evolution of Cooperation.* New York: Basic Books.

Axelrod, R., & Keohane, R. (1986). Avhieving Cooperation under Anarchy. In O. Kenneth, *Strategies and Institutions: Cooperation Under Anarchy* (pp. 226-254). Princeton, NJ: Princeton University Press.

Babbie, E. (1973). *Survey Research Methods.* , Belmont, California : Wadsworth Pub. Co.

Babbie, E. (2008). *The basics of social research,* . Belmont: Thomson Wadsworth.

Bagozzi, R. P., Yi, Y., & Phillips, L. W. (1991, September). Assessing Construct Validity in Organizational Research. *Administrative Science Quarterly* .

Banks, D., Erickson, J., & Rhodes, M. (2009, February 21). *Multi-tenancy in Cloud-based Collaboration Services.* Retrieved April 12, 2010, from Hewlett-Packard Development Company, L.P.: http://www.hpl.hp.com/techreports/2009/HPL-2009-17.pdf

bazar, B., & Boalch, G. (1997). A preliminary Model of Internet Diffusion within Developing Countries. *AUSWEB97 Conference.* Gold Coast.

benilian, A., Hess, T., & Buxmann, p. (2009). Drivers of SaaS-Adoption- An Empirical Study of Different Application Types. *SpringerLink- Business and Information Systems Engineering* , 357-369.

Bessen, J., & Hunt, R. (2007). An Empirical Look at Software Patents. *Social Science Research Network* , 157-189.

Bezroukov, N. (2002, March 9). *An Annotated Webliography on Open Source Software Development Problems.* Retrieved February 15, 2010, from Softpanorama.org: www,softpanorama.org/OSS/webliography.shtml

Bezroukov, N. (1990, October 4). Open Source Software Research as a Special Kind of Research: Critique of Vulgar Raymondism. *First Monday* .

Boettiger, S., & Burk, D. (2004). Open Source Patenting. *Social Science Research Network* , 221-231.

Bonaccorsi, A., & Cristina, R. L. (2003). Why Open Source Software Can Succeed . *Research Policy* , 1243-1258 .

Bonaccorsi, A., Rossi, C., & Giannangeli, S. (2004). Adaptive Entry Strategies under Dominant Standards-Hybrid Business Models in the Open Source Software Industry. *Social Science Research Network* .

Bouwman, H., Vos, H. D., & Haaker, T. (2008). *Mobile Service Innovation and Business Models.* Berlin: Springer-Verlag.

Brian, F. (2006, November 3). The Transformation of Open Source Software. *MIS Quarterly* , pp. 587-598.

Brodkin, J. (2008, July 2). *Gartner: Seven cloud-computing security risks.* Retrieved April 15, 2010, from InfoWorld: Security Central: http://www.infoworld.com/d/security-central/gartner-seven-cloud-computing-security-risks-853?page=0,1

Bruce, P. (2008, February 8). *State of Open Source Message: A New Decade For Open Source.* Retrieved October 24, 2009, from Bruce Perens: http://perens.com/works/articles/State8Feb2008/

Bruce, P. (2005). The Problem of Software Patents in Standards. In S. Bolin, *The Standards Edge: Open Season* (pp. 173-186). The Bolin Group.

Buyya, R., Ranjan, R., & Calheiros, R. (2009, May). *Modeling and Simulation of Scalabel Cloud Computing Environment and the CloudSim Toolkit> Challenges and Opportuinities.* Retrieved March 10, 2010, from ZDNet White papers: http://whitepapers.zdnet.com/abstract.aspx?docid=1198649

Buyya, R., Shin, C., Venugopal, S., Broberg, J., & Brandic, I. (2009). Cloud Computing and Emerging IT Platforms: Vision, Hype and Reality for Delivering Computing as the 5th Utility. *Portal* , 599-616.

Calpy, D., & Chenogorov, F. (2009). *Inhibitors to Open Source Software Adoption.* Jyväskylä.

Canepa, A., & Stoneman, P. (2004). Comperative International Diffusion: Patterns, Determinants and Policies. *Economic Innovation and New Technology* , 279-298.

Carlo, D., & Jesús, G.-B. (2000). *Free Software / Open Source:Information Society Opportunities for Europe?* Information Society Directorate General of the European Commission.

Carmines, E. G., & Zeller, R. A. (1991). *Reliability and viability assessment.* Thousand Oaks, Califonia: Sage Publication.

Carr, V. H. (1999, June 21). *Technology Adoption and Diffusion.* Retrieved November 8, 2009, from AWC Gateway web page: http://www.au.af.mil/au/awc/awcgate/innovation/adoptiondiffusion.htm

Chang, V., Mills, H., & Newhouse, S. (2007). Open Source to Long-term Sustainability: Review of Business Models and Case Studies. *OMII-UK Workshop.* Nottingham, UK: All Hands Meeting.

Cote, M. (2009, February 19). *Making Billions with Open Source, Revisited.* Retrieved February 10, 2010, from RedMonk: Cote's People over Process: http://www.redmonk.com/cote/2009/02/19/closedopensource/

Creswell, J. W. (2003). *Research design: qualitative, quantitative, and mixed method approaches.* Thousand Oaks, Carlifonia: Sage Publications Inc.

Csikszentmihalyi, M. (1990). *Flow: The Psychology of Optimal Experience.* New York: Harper & Row.

Daffara, C. (2009, July 1). *Rethinking OSS Business Model Classification by Adding Adopters' Value.* Retrieved November 26, 2009, from Free/Libre and Open Source Software Metrics, Conecta: http://guide.conecta.it/index.php/6._FLOSS-based_business_models

Daniel, C., & Chenogorov, F. (2009). *Inhibitors to Open Source Software Adoption.* Jyväskylä.

Dash, N. K. (2005, June). *Module: Selection of the Research Paradigm and Methodology.* Retrieved April 3, 2010, from Online Research Methods for Teachers and Trainers: http://www.celt.mmu.ac.uk/researchmethods/Modules/Selection_of_methodology/index.php

Dave, R. (2009, March 23). *Commercial open source, the future state.* Retrieved February 10, 2010, from cnet news: http://news.cnet.com/8301-13846_3-10202232-62.html

DiBona, C., & Ockman, S. (1999). *Open Source:Voices from the Open Source Revolution.* 101 Morris Street, Sebastopol, CA 95472.United Sates of America: O'Reilly Media.

Dixon, R. (2003). *Open Source Software Law.* Norwood, MA, USA: Artech House Incorporated.

Dubey, A. (2008, May 9). *Winning the SaaS Platform Wars.* Retrieved February 21, 2010, from SandHill.com: http://www.sandhill.com/opinion/editorial.php?id=188

Elena, B. (2009, September 7). *Dual Licensing.* Retrieved February 14, 2010, from OSS Watch: http://www.oss-watch.ac.uk/resources/duallicence2.xml

Evers, S. (2000, August 13). An Introduction To Open Source Software Development. Berlin, Germany.

Fichman, R. G. (1992). Information Technology Diffusion; A Review ofEmpirical Research. *13th International Conference on Information Systems ICIS* (pp. 195-206). Dallas: ICIS.

Fielding, R. T. (1999). Shared Leadership in the Apache Project. *Comminication of the ACM* , 42-43.

force.com cloud platform. (2009). *Force.com: A Comprehensive Look at the World's Premier Cloud-Computing Platform.* Retrieved March 10, 2010, from Salesforce.com corporate web site: http://www.developerforce.com/media/Forcedotcom_Whitepaper/WP_Forcedotcom-InDepth_040709_WEB.pdf

Fotescu, R.-C. (2007, April 14). *The sorry state of open source today.* Retrieved February 12, 2010, from The Jem Report: http://www.thejemreport.com/content/view/309/

Franke, N., & Hippel, E. v. (2003). Satisfying heterogeneous user needs via innovation toolkits: the case of Apache security software . *Research Policy* , 1199-1215 .

Gomulkiewicz, R. (2002). De-Bugging Open Source Software Licensing. *University of Pittsburgh Law Review* , 75-103.

Goode, S. (2004). *Something for Nothing: Management rejection of Open Source Software in Australia's top Firms.* Acton: Australian National University.

Gordijn, j., & Akkermans, H. (2001). Designing and Evaluating E-Business Models. *IEEE Intelligent Systems* , 11-17.

Gordijn, J., Akkermans, H., & Vliet, H. V. (2000). What's in an Electronic Business Model? *Knowledge Engineering and Knowledge Management - Methods, Models, and Tools, 12th International Conference* (pp. 257-273). Berlin: Springer-Verlag.

Gordijn, J., Osterwalder, A., & Pigneur, Y. (2005). Comparing two Business Model Ontologies for Designing e-Business Models and Value Constellations. *18th Bled Electronic Conference eIntegration in Action* (pp. 1-31). Bled, Slovania: MANAGEMENT INFORMATION SYSTEMS.

Greg, G. (2008). Software as a Service:The Spark That Will Change Software Engineering? *IEEE Distributed Systems Online , 9*, 3.

Griliches, Z. (1957). Hybrid Corn: An Exploration in the Economics of Technological Change. *Econometrica* , 501-522.

Grossman, R., & Gu, Y. (2009). On the Varieties of Clouds for Data Intensive Computing. *IEEE Computer Society* , 44-50.

Hamari, J., & Heikkila, J. (2010). Context- a Missing Bussiness Model Component?: Evidence from Virtual Goods Sales Business Model. *23rd Bled e Conference e trust.* Bled, Slovania.

Hancheng, L., & ChangQi, T. (2008). An Anatomy to SaaS Business Mode Based on Internet. *International Conference on Management of e-Commerce and e-Government* (pp. 215-220). JiangXi, China: IEEE Computer society.

Hecker, F. (1999). Setting up shop: The Business of Open Source Software. *IEEE Software* , 45-51.

Heller, M., & Eisenberg, R. (1998). Can Patents Deter Innovation? The Anticommons in Biomedical Research. *Science* , 698 - 701.

Herselman, M., Botha, G., & Brits, J.-P. (2007). Modeling Business Capabilities. *Proceedings of the 2007 Informing Science and IT Education Joint Conference* (pp. 151-170). Pretoria: Informing Science Institute.

Hertel, G., Niedner, S., & Herrmann, S. (2003). Motivation of Software Developers in Open Source Projects:an Internet-based Survey of Contributors to the Linux Kernel. *Research Policy* , 1159-1177.

Hoffer, J., & Alexander, M. (1992). The Diffusion of Database Machines. *ACM Portal, New York* , 13-19.

Hohensohn, H., & Hang, J. (2003). *Product- and Service Related Business Models for Open Source Software.* Furstenallee, Paderborn: Siemens Business Services GMBH & Co. OHG C - LAB.

Hoover, N., & Martin, R. (2008). Demystifying the Cloud. *InformationWeek Research & Reports* , 30 -3 7.

Iansiti, M., & Richards, G. (2006). *The Business of Free Software: Enterprise Incentives, Investment, and Motivation in the Open SourceCommunity.* Cambridge, Masachusets: HBS (Harvard Business School).

Jason, S. (2008, March 24). *The state of open source: Issues and opportunities.* Retrieved February 12, 2010, from Developer World-InfoWorld: http://www.infoworld.com/d/developer-world/state-open-source-issues-and-opportunities-416?page=0,2&_kip_ipx=480865819-1269649613&_pxn=0

Judd, C. M., & Kenny, D. A. (1981). *Estimating the effects of social interventions.* New York: Cambridge University Press.

Jullien, N., & Zimmermann, J.-B. (2007). Free/Libre/Open Source Software (FLOSS): lessons for intellectual property rights management in a knowledge-based economy. *Social Science Research Network* , 19-36.

Jullien, N., & Zimmermann, J.-B. (2006). New Approach to Intellectual Property:From Open Software Knowledge Based Industry. *System Design Frontier* , 33-56.

Karels, M. (2003). Commercializing Open Source Software. *Association for Computing machinery, acmqueue* , 47-55.

Karim, L., & Robert, W. (2003). Why Hackers Do What They Do:Understanding Motivation and Effort in Free/Open Source Software Projects. *Social Science Research Network (SSRN eLibrary)* .

Keiran, H., & Schussman, A. (2003). *The Ecology of Open-Source Software Development.* Tucson: University of Arizona.

Kern, T., Kreijger, J., & Willocks, L. (2002). Exploring ASP as Sourcing Strategy: Theoritical Perspective, Propositions for Practice. *Strategic Information Systems* , 153-177.

Kern, T., Lacity, M., & LWillocks. (2002). *Netsourcing: Renting Business Applications and Services Over a Network.* New York: Prentice Hall.

Kim, W. (2009). Cloud Computing: Today and Tomorrow. *Journal of object technology* , 65-72.

Koenig, M. (2006). Configurability in SaaS (Software as a Service) Applications. *ACM Portal* , 19-26.

Kogut, B., & Metiu, A. (2001). Open Source Software Development and Distributed Innovation. *Oxford Journal* , 248-264.

Koski, H. (2008). *Do Technological Diffusion Theories Explain the OSS Business Adoption Patterns?* Helsinki: ETLA, The Research Institute of the Finnish Economy.

Koski, H., & Kretschmer, T. (2007). *Competing in Network Industries: Firm strategies, Market Outcomes and Policy implications.* London: London School of Economics and ETLA.

Krishnamurthy, S. (2003, September). A managerial overview of open source software. *Business Horizons , 46* (5), pp. P.47-56.

Krishnamurthy, S. (2005, January 18). *An Analysis of Open Source Business Models*. Retrieved November 23, 2009, from Social Science Research Network: http://www.ssrn.com/paper=650001

Krogh, G. v., Spaeth, S., & Lakhani, K. R. (2003). Community, Joining, and Specialization in Open Source Software Innovation: A Case Study. *Research Policy* (32), 1217-1241.

Kumiyo, N., Yasuhiro, Y., Yoshiyki, N., Kouichi, K., & Yunwen, Y. (2002). *Evolution Patterns of Open Source Software Systems and Communities.* Tokyo: Japan Society for the Promotion of Science.

Lai, V., & Mahapatra, R. (1997). Exploring the Research in Information Technology Implementation. *Elsevier Science Publishers* , 187-201.

Lambert, S. (2008). A Conceptual Framework for Business Model Research. *21st Bled eConference eCollaboration:Overcoming Boundaries through Multi-Channel Interaction.* Bled, Slovania.

Lampitt, A. (2008, August 29). *Open-Core Licensing (OCL): Is this Version of the Dual License Open Source Business Model the New Standard?* Retrieved February 15, 2010, from Lampitt or Leave It: http://alampitt.typepad.com/lampitt_or_leave_it/2008/08/open-core-licen.html

Lanzara, G. F., & Morner, M. (2003). The Knowledge Ecology of Open-Open Source Software Project. *European Group of Organisational Studies.* Copenhagen.

Lazarsfeld, P., & Katz, E. (1955). *Personal Influence.* New Yorl: The Free Press.

Lehman, S. (2008). *Revenue and Pricing Models for Software Vendors with a Focus on Software as a Service.* Darmstadt, Germany: SAP Research CEC.

Lerner, J., & Tirole, J. (2002). Some Simple Economicsof Open Source Software. *Journal of Industrial Economics* , 197-234.

Lerner, J., & Tirole, J. (2000). The Simple Economics of Open Source . *Social Science Research Network* .

Lindenberg, S. (2001). Intrinsic motivation in a new light. *Kyklos* , 317-342.

Linus, T., & David, D. (2001). *Just for Fun. The Story of an Accidental Revolutionary.* New York: HarperCollins Publishers Inc.

Madanmohan, T. R., & Pal, N. (2002). *Competing on Open Source: Strategies and Practise.* Retrieved October 09, 2009, from COSPA Knowledge Base: http://hdl.handle.net/2038/1092

Malcom, J. (2002). *Problems in Open Source Licensing.* Retrieved March 2, 2010, from iLaw, Barristers and Solicitors: http://www.ilaw.com.au/public/licenearticle.html#sdfootnote2sym

Malik, O. (2006, July 7). *Coghead, A New Web App Machine.* Retrieved January 12, 2010, from GIGAOM: http://gigaom.com/2006/07/07/coghead-a-new-web-app-machine/

Mansfield, E., Romeo, A., Schwartz, M., Teece, D., & Wagner, S. (1983). New Findings in Technology Transfer, Productivity and Economic Policy. *Research Management* , 11-20.

Marshall, C., & Rossman, G. (2006). *Designing Qualitative Research.* London: Sage Publication Inc.

Mathew, S. (2009, April 14). *The Modern State of Software Innovation: How Sun and Oracle Are Changing Their Open Source Diet.* Retrieved January 3, 2010, from InformIT: http://www.informit.com/articles/article.aspx?p=1336903

Mckinsey and Company. (2008). *Emerging Platform Wars in Enterprise Software.* Newy york: Mckinsey and Company Inc.

McKusick, M. K. (1999). Twenty Years of Berkeley Unix:From AT&T-Owned to Freely Redistributable . In C. DiBona, & S. Ockman, *Open Sources: Voices from the Open Source Revolution* (pp. 31-46). 101 Morris Street, Sebastopol, CA : O'Reilly & Associates, Inc, .

Mell, P., & Grance, T. (2009, October 7). *The NIST Definition of Cloud Computing.* Retrieved February 18, 2010, from NIST: National Institute of Standards and Technology: http://csrc.nist.gov/groups/SNS/cloud-computing/

Metz, S., Eschinger, C., Pang, C., & Dharmasthira, Y. (2008). *User Survey Analysis: Software as a Service, Enterprise Application Markets.* Worldwide: Gartner Inc.

Miller, M. (1987). *Miller, M. (1987): Accounting theory and policy: a reader.* Orlando, Florida: Harcourt Brace Jovanovich Inc.

Ming-Wei, W., & Ying-Dar, L. (2001). Open Source Software Development: an Overview. *IEEE,Computer* , 33-38.

Moore, G., & Benbasat, I. (1991). Development of an Instrument to Measure the Perceptions of Adopting an Information Technology Innovation. *Information System Research* , 192-222.

Motahari-Nezhad, H. R., Stephenson, B., & Singhal, S. (2009). *Outsourcing Business to Cloud Computing Services: Opportunities and Challenges.* Palo Alto, CA, USA: Hewlett Packard Labs.

Murray, G. F., & Duncheon, M. (2006, January). *FREE AND OPEN SOURCE SOFTWARE:An Introduction.* Retrieved February 12, 2010, from open-bar.org/: http://www.open-bar.org/docs/Free-and-Open-Source.pdf

Muselli, L. (2002). Licenses: Strategic Tools for Software Publishers_. *New Software Industry Economy* , 129-145.

Myers, M. (1997). "Critical Ethnography in Information Systems," in Information Systems and Qualitative Research, London. *Chapman & Hall, Ltd. London,* , 276-300.

Nahar, N. (2001). Diffusion of Software Engineering Innovation in the Global Context. *Proceedings of the 24th information Systems Research Seminar in Scandinavia* (pp. 655-664). Ulvik: R.E. Moe, A.I. Morch and A.L. Opdahl.

Nemiro, J. (2002). The creative Process in Virtual Teams. *Creative Research Jurnal* , 69 - 83.

O'Donoghue, T., & Punch, K. (2003). *Qualitative Education in Action: Doing and Reflecting.* New York: RoutledgeFalmer.

O'Leary-Kelly, S. W., & Vokurka, R. J. (1998). The empirical assessment of construct validity . *Journal of Operations Management* , 387-405.

Orr, G. (2003, March 18). *Diffusion of Innovations, by Everett Rogers (1995) Review.* Retrieved March 18, 2010, from http://www.stanford.edu/class/symbsys205/Diffusion of Innovations.htm

Osterwalder, A. (2004). *The Business Model Ontology - a proposition in a design science approach.* lausane: University of Lausanne.

O'Sullivan, M. (2002). Making Copyright Ambidextrous: An Expose of Copyleft. *Journal of Information, Lawand Technology* .

Pavlicek, R. (2000). *Embracing Insanity: Open Source Software Development* . Indianapolis: Sams Publishing.

Pettey, C. (2006). *Gartner Says 25 percent of New Business Software will be Delivered as Software as a Service by 2011.* Stamford: Gartner.

Premkumar, G., & Potter, M. (1995). Adoption of Computer Aided Software Engineering (CASE) Technology: An Innovation Adoption Perspective. *ACM Portal, New York* , 105-124.

Raymond, E. (2001). *The Cathedral and the Bazaar:Musings on Linux and Open Source by an Accidental Revolutionary.* 101 Morris Street,Sebastopol, CA95472.United states of America: O'Reilly and Associates Inc.

Riehle, D. (2007). The Economic Motivation of Open Source Software: Stakeholder Perspectives . *IEEE COmputer* , 25-32.

Rivette, K., & Kline, D. (2000). *Rembrandts in the attic: unlocking the hidden value of patents.* Cambridge: Harvard Business Press.

Rogers, E. (1983). *Diffusion of Innovation* . New York: The Free Press.

Rogers, E. (1995). *Diffusion of Innovation.* New York: Free Press.

Rosen, L. (2004). *Open Source Licensing: Software Freedom and Intellectual Property Law.* New York: Prentice Hall.

sainio, L.-M., & Marjakoski, E. (2009). The Logic of Revenue Logic? Strategic and Operational Levels of Pricing in the Context of Software Business. *ScienceDirect* , 368-378.

Sala, E., & Lynn, P. (2009). The Potential of a Multi-mode Data Collection Design to Reduce non-response bias. The Case of a Survey of Employers. *Quality and Quantity* .

Saloner, G., & Shepard, A. (1995). Adoption of Technologies with Network Effects: An Empirical Examination of the Adoption of Teller Machines. *Rand Journal of Economics* , 479-501.

Sarkinen, J. (2007). An Open Source(d) Controller. *IEEE Xplore* , 761-768.

Scacchi, W. (2004). Socio-Technical Interaction Networks in Free/Open Source Software Development Processes. *Springer Science* , 1-27.

Scott, S., Plotnikoff, R. C., Karunamuni, N., Bize, R., & Rodgers, W. (2008, October 2). *Factors influencing the adoption of an innovation: An examination of the uptake of the Canadian Heart Health Kit (HHK).* Retrieved March 4, 2010, from Implementation Science: http://www.implementationscience.com/content/3/1/41

Seppä, A. (2006). *Open Source in Finnish Companies.* Helsinki: Etla.

Smets-Solanes, J., & Faucon, B. (1999). *Logiciels Libres-Liberte, Egalite,Business.* Paris: Edispher.

Smith, R. (2010, March 19). *Government Applications of Cloud Computing.* Retrieved April 20, 2010, from Model Benders, LLC: http://www.modelbenders.com/papers/RSmith_Cloud_HPTi.pdf

Stefan, C., Fabio, M., & Maria, L. (2007). From Planning to Mature:on the Success of Open Source Projects. *Science Direct* , 1575-1586.

Subramanian. (2008). *Open Source Value Addition in SaaS* . Retrieved from http:

Subramanian, K. (2008, December 01). *Open Source Value Addition in SaaS.* Retrieved December 28, 2009, from CloudAVE web site: http://www.cloudave.com/link/open-source-value-addition-in-saas

Sun Microsystems,Inc. (2009, July). *OPEN SOURCE IN THE ENTERPRISE:FULFILLING THE PROMISE.* Retrieved December 23, 2010, from Sun Microsystem Corporation Website: https://www.sun.com/offers/docs/opensource_enterprise.pdf

Tessmer, M. (1990). Environmental Analysis: A Neglected Stage of Instructional Design. *Educational Technology Research and Development* , 55-64.

The Guardian. (2008, April 17). *Google angles for business users with 'platform as a service'.* Retrieved March 21, 2010, from guardian.co.uk: http://www.guardian.co.uk/technology/2008/apr/17/google.software

Timmers, P. (1998). Business Models for Electronic Markets. *Electronic Markets* , 3-8.

Torvalds, L. (1999). The Linux Edge. *Communications of the ACM , 42 number 4*, 38-39.

Tsikriktsis, N., Lanzolla, G., & Frohlich, M. (2004). Adoption of e-processes by Service Firms: An empirical study of antecedents. *Production and Operations Management Society* , 216-229.

T-Systems. (2008). *CloudComputing.AlternaticeSourcing Strategy for Business ICT.* frankfurt: T-Systems Enterprise Services GmbH.

Tucker, L. (2010). Introduction to Cloud Computing for Enterprise Users. *Cloud Computing Conference.* Silicon Valley,CA: Sun MicrosystemInc.

Turner, M., Budgen, D., & Brereton, P. (2003). Turning Software into a Service. *IEEE Computer Society* , 38-44.

Ueda, M. (2005). Licenses or Open Source Software and thier Economic Value . *2005 Symposium on Applications and the Internet Workshop (SAINT-W'05).* IEEE Computer Society.

valimaki, M. (2006). Copyleft Licensing and EC Competition Law. *Social Science Research Network* , 130-136.

Wang, L., & Laszewski, G. V. (2008). *Cloud Computing:A Perspective Study.* Rochester, NY: Service Oriented Cyberinfrastructure lab,Rochester inst. of Tech.

Webb, E., Campbell, D., Schwartz, R., & Sechrest, L. (1966). *Unobtrusive Measures: Nonreactive Research in the Social Science.* Chicago: Rand McNally.

Weber, S. (2004). *The Success of Open Source.* Cambridge, massachusetts: Harvad University press.

Weis, A. (2007). Computing in the Clouds. *ACM, New York* , 16-25.

Wichmann, T. (2002). *Basics of Open SourceSoftware Markets and Bussiness Models. Free/Libre and Open Source Software:Survey and Study, FLOSS Final report.* Maastricht, netherlandsand Berlin Germany: International Institute of Infonomics, Berlecom Research GmbH.

Williamson, O. (1975). *Markets and Hierarchies: Analysis and Antitrust Implications.* New York: Free Press.

Wilson, R. (2009, October 05). *Software Patents.* Retrieved March 02, 2010, from OSS-Watch: http://www.oss-watch.ac.uk/resource/softwarepatents.xml

Von Krogh, G., Kazou, I., & Ikujiro, N. (2000). *Enabling Knowledge Creation.* New York: Oxford University Press.

VonHippel, E. (2005). *Democratizing Innovation.* Cambridge: MIT Press 5 Cambridge center.

Vouk, M. (2008). Cloud Computing- Issues, research and Implemetations. *IEEE Xplore* , 31-40.

Välimäki, M. (2001). Strategic Use of Intellectual Property Rightsin Digital Economy- Caseof Software Markets. *4th International Conference on the Management of Intellectual Capital and Electronic Commerce.* Hamilton, Canada.

Välimäki, M. (2005). *The Rise of Open Source Licensing. A Challenge to the Use Of Intellectual Property in the Software Industry.* Helsinki: Turre Publishing.

Yeats, D. (2008). The Role for Technical Communicators in Open-Source Software Developmen. *Technical Communication* , 38-48.

Yyn, R. (1994). *Case Study Research: Design and Methods.* Beverly Hills,California : Sage Publications.

APPENDIXE 1: THE INTERVIEW QUESTIONNAIRE GUIDE

1) INTERVIEW WITH IPR EXPERT

General background of the interviewee

1. Can you tell me a bit about your background, company and job title
2. When was your company established
3. In which industry does your company carry out its principal operations
4. Is your company involved with open source software activities? What, how and in what areas?
5. How long have you been working in the field of software development?

Examining Open source software issues

6. How would you describe your companies OSS business model with respect to the following examples?
   - Support and service contract
   - Split/Dual licensing
   - Value added closed source
   - Hardware integrators
   - Open source distributors and platform providers
7. What in your opinion are the main attractions for companies to (a) produce software using the open source approach and (b) adopt and implement open source instead of proprietary software?
8. What in your opinion are the major deterrents for a company to adopt OSS or develop and license software as open source?

Examining OSS as a service

9. Have you ever considered the possibility of including OSS components as part of a SaaS offering?
10. What are the major advantages of this approach?
11. What are the limitations of this approach

12. What additional legal obligations obligation does the inclusion of open source software as part of a SaaS offering impose on the users and providers of the service?
13. What additional legal obligations obligation does the inclusion of OSS as part of a PaaS offering impose on the consumers and providers of the service?

2) INTERVIEW WITH SOFTWARE APPLICATION DEVELOPERS

Introduction

1) What industries do you work in?
2) What is the size of your company? – employee or financial size –
3) Does your company develop/ use web based applications?

Use of open source software

4) Does your company use open source software?
5) What kind of open source software product/ component does your component use?
6) If possible could you elaborate on what you use them for and how they are integrated to your core activities?
7) What typical problems do you face with it (technical, legal, financial, acquisitioned etc)?

Cloud Computing

8) Are you familiar with the term cloud computing?
9) What kind of cloud based service is/would be useful for your organisation?
10) What kind of services would you prefer in a cloud platform as a service?
    - Development and testing platform
    - Hosting integrating and deploying software application
    - Security, source code and version control services
    - Provision of mash up services with open source software components
    - Workflow facilities for application design.
11) How would the inclusion of OSS components in the service offering mix affect your decision making process?
12) What factors encourage you to take up a PaaS offering
    - Reduction in upfront cost (capex)
    - PaaS increases flexibility of adopter company
    - Allows the company to focus on its core business

- Allows organisations to be agile and responsive to the market
- Ability to immediately tap into computing power and software
- Facilitates access to the latest technology

13) What fears and worries do you have about outsourcing to the PaaS model?
    - Cost
    - Security concerns (which ones)
    - Performance (how)
    - Scalability
    - Availability
    - Integration with existing IT system
    - Vendor lock-in
    - Loss of control over service/data
    - IPR issues

14) What do you think about the future of cloud computing and specifically web based application development platform as a service and do you think there is a place for open source software in this new paradigm?

APPENDIX 2 ONLINE SURVEY FORM

1. Introduction

1. What industry is your company in?

- ○ Finance
- ○ Energy
- ○ Media
- ○ Telecommunications
- ○ Manufacturing
- ○ Healthcare
- ○ Software service/product development
- ○ Government / Public Sector Other

Other (please specify) ____________

2. How many employees does your company have

- ○ 0-10
- ○ 10-20
- ○ 20-50
- ○ 50-100
- ○ >100

2. Awareness and perception of PaaS

1. Are you familiar with the term Cloud computing?

○ Yes

○ No

Cancel Cop

2. If you selected Yes in question 3 do you think that the role of cloud based services will increase in the next five years?

○ Yes

○ No

3. Do you think that the role of cloud based open source software components will increase in the next five years

○ Yes

○ No

3. Adoption and inhibitors of PaaS

1. Which layer(s) of the Cloud have you adopted or are most likely to adopt? (Please select as many options as are relevant).

☐ Individual software packages (SaaS).

☐ Complete operating system, development platform and software packages available via cloud services (PaaS)

☐ Infrastructure services such as storage, network capacity etc (Iaas)

☐ No intention to adopt any cloud technology

☐ I don’t know

2. If you selected PaaS in question 1 above, what type of Cloud Platform as a service offering has your organization adopted/is likely to adopt (please select as many options as are relevant).

☐ Development and testing platform

☐ Hosting, integrating and deploying software application

☐ Files and database hosting

☐ Security, source code and version control services

☐ Provision of Mashup services with Open source components

☐ Workflow facilities for application design

Other (please specify)

3. How will the inclusion of open source components in the PaaS offering mix affect you/your company's adoption decision process?

○ Strong negative influence

○ Negative influence

○ No effect

○ Positive influence

○ Strong positive influence

4. Do you agree with the perceived benefits below of adopting Platform as a Service technology?

	Yes I agree	**No I don't agree**	**I don't Know**
PaaS Reduces upfront costs	○ I agree	○ No I don't agree	○ I don't Know

(Capital expence)			
PaaS Increases flexibility of the adopter company	○ Yes I agree	○ No I don't agree	○ I don't Know
PaaS Allows the company to focus on it's core business	○ Yes I agree	○ No I don't agree	○ I don't Know
PaaS enables its adopters to be agile and responsive to the market.	○ Yes I agree	○ No I don't agree	○ I don't Know
PaaS facilitates access to the latest technologies	○ Yes I agree	○ No I don't agree	○ I don't Know
PaaS provides the ability to immediately tap computing power and software	○ Yes I agree	○ No I don't agree	○ I don't Know

5 In your opinion what are the main inhibitors to the PaaS adoption? (Please select how important the element is from the list and select as many options as are relevant).

	Very important	**Important**	**Slightly important**	**Not important**
Cost	Very important	○ Important	○ Slightly important	○ Not important
Security concerns	○ Very important	○ Important	○ Slightly important	○ Not important
Performance	○ Very important	○ Important	○ Slightly important	○ Not important

Scalability	○ Very important	○ Important	○ Slightly important	○ Not important
Availability	○ Very important	○ Important	○ Slightly important	○ Not important
Integration with existing IT system (results in difficulties to migrate to the clouds)	○ Very important	○ Important	○ Slightly important	○ Not important
Privacy issues	○ Very important	○ Important	○ Slightly important	○ Not important
Vendor lock in	○ Very important	○ Important	○ Slightly important	○ Not important
Loss of control over service/data	○ Very important	○ Important	○ Slightly important	○ Not important
IPR issues	○ Very important	○ Important	○ Slightly important	○ Not important

Appendix 3 List of Online forums

1) Cloud Computing forum
2) Cloudcomputing forum
3) Azure Talk- Windows cloud computing platform discussion forum
4) Cloud Standards forum
5) Cloud Computing Virtualization, OS, Web 3.0, Unified computing forum
6) Cloud Hosting and service providers forum
7) Cloud Computing Interoperability forum (CCIF)
8) Cloud Mobility 2010
9) Oracle cloud forum
10) Cloud Computing world forum
11) GID forum- Computer software forum
12) SAP Community network
13) Fedora forum
14) Tech- forum
15) Forum Nokia