FH Technikum Wien

MASTER THESIS

submitted in partial fulfilment of the requirements for the degree of

„Master of Science in Engineering“

Open Source Software in strategic IT management:
Application and strategic economic contribution of Open Source Software to business success

by Wolfgang Rogner

A-2351 Wr. Neudorf, Reisenbauerring 1/5/23

Main Advisor: DI.DDr. Walter Jaburek

Co-advisor: Mag. Edmund Humenberger

Wr. Neudorf, 15.5.2011

FACHHOCHSCHULE TECHNIKUM WIEN

Eidesstattliche Erklärung

„Ich erkläre hiermit an Eides Statt, dass ich die vorliegende Arbeit selbständig angefertigt habe. Die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht (Anführungszeichen, kursive Schrift, im Originaldokument als eigener Schriftstil definiert). Eigenleistungen wurde grau unterlegt. Die Arbeit wurde bisher weder in gleicher noch in ähnlicher Form einer anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.“

Acceptance of liability

„I declare that this thesis was written completely by myself. References to external sources (including my own previous work) have been denoted explicitly (quotation marks text in italic, unique style sheet in the original document). Own contributions are shaded grey. This paper has not been published in part or full previously or submitted to a different board.“

Ort, Datum	Unterschrift

Kurzfassung

Die vorliegende Arbeit beschreibt die Situation in welcher sich IT Abteilungen und IT Manager aufgrund globalisierter, gesättigter Märkte und steigendem Erfolgsdrucks auf Unternehmen befinden. Die Ursachen für immer geringere Nutzenbeiträge der IT zum Gesamtunternehmen werden identifiziert. Open Source Software (OSS) wird als möglicher Ausweg aus dem Dilemma sinkender IT-Budgets und steigender Geschäftsanforderungen angeboten. Die Arbeit untersucht den strategischen Nutzenbeitrag von OSS zum Unternehmen aus wirtschaftlicher Sicht. Sie begründet sich auf Literaturrecherche öffentlich zugänglicher Informationen, Beobachtungen und dem Vergleich von Fallbeispielen. Ergebnisse wurden auf Basis des Grounded Theory Ansatzes entwickelt und argumentiert. Die Arbeit stellt strategische Nutzenbeiträge von OSS im betrieblichen Umfeld dar, fasst eine Liste kritischer Erfolgsfaktoren zusammen und leitet daraus Perspektiven und Messkriterien für den Einsatz in einer IT-Balanced Scorecard ab. Wesentliche Nebenergebnisse sind die langfristige Dominanz von OSS sowie Aussagen zum optimalen Umfeld für den Pioniereinsatz.

Schlagwörter: Open Source Software, Strategisches IT Management, Strategischer Nutzenbeitrag, Unternehmenspotentiale, IT-Balanced Scorecard

Abstract

This thesis describes the situation IT departments and IT managers face due to globalised markets and increasing cost pressure. It discusses reasons for the diminishing contribution of IT to the overall enterprise profit. Open Source Software (OSS) is offered as one solution to the dilemma of reduced IT budgets and increasing demands by company leaders. The paper scrutinizes the strategic benefits of using OSS from an economical point. Relevant information was derived from open literature, observation and comparison of use cases. Results were developed based on grounded theory. This paper summarises the strategic benefits of OSS and its economic contribution, provides a list of critical success factors for implementing OSS within companies and develops a set of perspectives and parameters for use in IT-Balanced Scorecard monitoring. Important secondary results are the long-term dominance of OSS as well as optimal preconditions for trailblazing implementation.

Keywords: Open Source Software, Strategic IT Management, Strategic benefit, Business opportunities, IT-Balanced Scorecard

For Michi and Paula

and my parents

Thanks to my advisors

Honourable mention:

Ing. Philipp Stiegler

This document and all related material was created using open source software

Table of Content

1 Introduction....................1
1.1 Problem statement....................4
1.2 Research questions....................7
1.3 Methodical approach....................8
1.4 Research proceedings....................8
1.5 Assumptions, requirements and preconditions....................10
1.6 Expected results....................11
1.7 Document structure....................11
2 Research status....................13
2.1 Strategic IT management....................13
2.2 Strategic benefits of Open Source Software....................18
2.2.1 Vendor independence....................24
2.2.2 Availability of source code....................25
2.2.3 Interoperability....................27
2.2.4 Efficient use of resources....................28
2.2.5 Other factors mentioned....................31
2.2.6 New potential strategic benefits....................37
2.3 Critical success factors....................42
2.3.1 Fraunhofer Institute....................42
2.3.2 Optaros....................44
2.3.3 Österreichischer Städtebund....................44
2.3.4 Dahlander and Magnusson on communities....................46
2.3.5 Vienna's OSS migration....................47
2.4 Metrics and key performance indicators....................49
2.4.1 Financial indicators....................50
2.4.2 Quality indicators....................53
2.4.3 Environmental indicators....................57
2.4.4 Internal indicators....................59
3 Concepts....................62
3.1 Motivation....................62
3.2 Definitions....................63
3.2.1 Strategic IT Management....................63
3.2.2 Up-stream development....................63
3.3 Conclusions from Chapter 2....................64
3.3.1 Strategic benefits of up-stream development....................64
3.3.2 OSS as a strategic tool to substitute CSS....................66
3.3.3 Strategic impact of OSS, results from the SWOT analysis....................70
3.3.4 Application of strategic benefits to TOWS matrix....................70

3.3.5 Potential users of OSS....................73
3.4 OSS value proposition....................77
3.5 Preconditions for OSS implementation....................82
3.5.1 Preconditions for the introduction and implementation of OSS....................83
3.5.2 Preconditions for the adoption and business continuity phase of OSS......85
3.6 Metrics....................88
3.7 Economic benefits....................90
4 Critical evaluation....................92
5 Conclusion....................93
Bibliography....................96
Illustrations....................98
Tables....................99
Abbreviations....................100
Appendix A: SWOT Analysis of Open Source Software....................101

1 Introduction

In its December 2009 Global CIO Report (Ferraris P., Porter M., 2009) Capgemini concludes:

> *"2009: IT-budgets shrink, expectations grow*
>
> *The 2009 economic downturn had a significant impact on IT budgets, with almost three quarters of CIOs (70%) reporting a decrease. On average, IT budgets dropped by 15%. Perhaps more surprisingly, CIOs say they are using the crisis to show the value of IT for their companies, by giving priority to projects that contribute more to the business or taking advantage of new market conditions."*
>
> *(Ferraris P., Porter M., 2009)*

Tight IT budgets are nothing new. While CISCO's COO John Chambers announced strong quarterly earnings in 2007, enraged users commented on tight budget situations on the internet :

> *"I'm under a lot of budget pressure by the board, however at the same time I'm not allowed to do any innovation. I must use "brand name" systems with off the shelf software configuration but cannot use any "unproven" technology ..."*
>
> *(mikey3211, Techrepublic, 2007)*

This implies that IT budgets are spent on maintaining existing (legacy) systems rather than investing into new technology and solutions. Going back further, EDUCAUSE centre for applied research in their "*ECAR Research Study 7*" summarised in their key finding:

> *"Key findings:*
>
> - *Two-thirds of respondents are under pressure to reduce their IT costs.*
> - *No single cost-reduction strategy predominates, but the most frequently used strategy is across-the-board cuts.*
> - *Among the numerous concerns about outsourcing, the greatest reason it is not being pursued is institutional culture.*
> - *There is a growing interest in shared IT services between institutions as a means of controlling costs.*

- *While most IT organizations are searching for new revenue sources, these will likely have only a marginal impact on most institutions' total IT budgets."*

(EDUCAUSE, 2004)

This indicates the pressure to achieve more with less (means). Anticipated benefits of IT outsourcing projects began to vanish or were at least questioned considerably due to risen communication and cost for error correction. Efforts to off-shore or near-shore IT were not satisfactory. There is a tendency to reverse these advances. The report also indicates that businesses are not willing to attribute generated rises of revenue due to increased corporate business opportunities to IT invocation and thus deny to share revenue with IT for further investment. In other words CIO's cannot prove that IT contribute to overall corporate revenue and companies do not want to share.

The "*Global CIO Report 2009*" argues:

> *"**IT usage as a business value enhancer***
>
> *We found IT functions tend to focus on the deployment of IT systems up until the moment the project goes live. But the value does not stop on the go-live date but starts on it. ...*
>
> *... CIOs surveyed confirm what academic research indicates: 80% of the value of IT is driven by usage, whereas technology deployment itself only accounts for 20%."*
>
> *(Ferraris P., Porter M., 2009)*

The perception of IT within and its value for the company is mainly driven by the cost up to the moment of deployment. Acknowledgement of the benefit and value of IT for companies largely depends on the dexterity of IT representatives marketing IT services and solutions.

2006, Forrester Research introduced their IT archetype pyramid, dividing IT into three main types:

- *"Solid Utility"*: providing proven, cost-effective technology (commodity) with an emphasis on declining cost
- *"Trusted Supplier"*: providing solid utility services plus extended IT project management including in-time, on-budget projects and applications. Services offered are driven by SLA's. Extracting business value lies within the responsibility of the business owner
- *"Partner Player"*: provides competitive solutions for customers, suppliers and the business on top of trusted supplier duties. Solutions are co-developed with the

business owner, success is measured in terms of company performance

These archetypes may be considered as different levels of trust, the company puts into its IT department. Each level requires the IT to profoundly service the demands of lower lying levels. Maintaining a certain level of competence requires ongoing effort and IT marketing. Aspiring for the next level requires extensive effort and service quality. There is no guarantee for the IT department to reach the aspired level. In certain situations it seems easier for external consultants to act as "Partner Player" than for internal IT departments. The extra cost for this type of consulting usually is subtracted from IT budgets.

In 2009, Capgemini in its "*Global CIO report*" extended this perspective with survey-based numbers:

- 24% of surveyed companies viewed their IT as being "*Technology Utility"* (relating to Solid Utility in Forresters pyramid)
- 39% saw their IT being a "*Service Centre"* (which equals to Forresters Trusted Partner)
- 37% finally considered their IT being a valuable "*Business Technology Partner"* (being Forresters Partner Player)

The report cites that "*IT wants to get closer to the business not because of the economy but because it's the right thing to do*[1]*. (CIO from the UK)*" (Ferraris P., Porter M., 2009).

While from an IT point of view this seems the natural progress and aspiration, companies tend to see IT as a commodity that just needs to work. This leads to diametric interests of business and IT. Metaphorically, IT constantly has to swim against the current that drags it down to providing commoditised utility at reduced budgets. However, interesting business challenges and larger financial funds lie at the business technology layer in Forresters pyramid model.

This raises the question: "*What is/are the right thing/things to do?*"

According to Tiemeyer, E. 2010, this is the core question of strategic IT management. Tiemeyer's model for strategic IT management is based on the model for strategic management of enterprises as described by Gälweiler A., Malik F., 2005. Both models see the primary purpose of strategic (IT) management in

- preserving existing potentials and
- establishing new potentials

to support ongoing corporate success. Strategic IT management theory will be dealt in more detail in chapter 2.1.

1 This statement was neither motivated nor explained in the report

In 2003, University St. Gallen carried out a study on the strategic impedance of open source software in Switzerland (Maas W., 2003). The study reveals (not surprisingly) that in considering total cost of ownership (TCO) and return on investment (ROI) issues, open source, specifically Linux, helps reduce cost and consolidate IT infrastructure.

Specific savings can be associated to:

- license fees
- operational costs
- costs for IT infrastructure
- support and maintenance fees
- cost for training

The study identified two specific trends: Linux entering the midrange market and Linux being introduced to the host by IBMs open source activities[2].

Maas's study suffered from a limited set of samples. However, it indicated the importance of open source as a strategic option for IT management.

1.1 Problem statement

IT has always faced restrictive environmental conditions.

Ongoing globalisation introduced international business relations requiring IT to provide 24 hour long distance communications and services. Strong competition from low-cost economically emerging countries like CEE and far east Asia increase pressure on time frames and IT budgets.

On the supply side, IT departments meet consolidated and monopolised markets which are adamant with respect to price or technology, sometimes even dictating the pace of technological advances. With multi-layer distribution channels[3], vendors decouple themselves from the responsibilities of fulfilment and warranty.

Internally, IT departments face progressive production cost. Extended automation, growing complexity of IT systems and interfaces require highly trained and skilled personnel. This continually raises labour cost while there is no obvious increase in productivity. Eventually this reduces the contribution IT adds to the overall company

2 At the time, this survey was conducted, IBM's open source strategy was not as diversified as it is at the time this paper was written. Today, IBM offers a range of open source related services, ranging from open source software development, consulting, computing center operations and specifically adapted open source operating systems for it's hardware line

3 Multi-layer distribution channels: Vendor, distributors, resellers and solution partners located in different geographical and national areas with different legal foundation make it hard to claim resolution of problem issues and redeem warranties from a single responsible instance

profits.

A natural impulse by top management is to cut down IT budgets. At first glance this will force IT departments to be more creative to achieve the same results or better. Unfortunately this is not the case. Maintenance cost for installed and established IT systems is fixed by contract and cannot be reduced. Turning off running IT systems is hardly an option, the only margin available to budget cuts is personnel, empowerment and innovation.

Efforts to cut down on IT expenses are widespread: Virtualisation of hard- and software, outsourcing of non-business-critical IT operation and off-shoring of subordinate IT activities to low-priced countries are valuable candidates for cost cutting.

Moderate development or even reduction of IT costs in outsourcing projects depends to a large extent on stable quantities and requirements. Changes in these parameters during outsourced operation usually lead to increased operational expenses. To prevent such a development, constant and accurate monitoring of critical operational parameters is due. This ongoing monitoring, efforts to maintain internal consolidation as well as necessary changes neutralise some of the savings achieved by outsourcing IT.

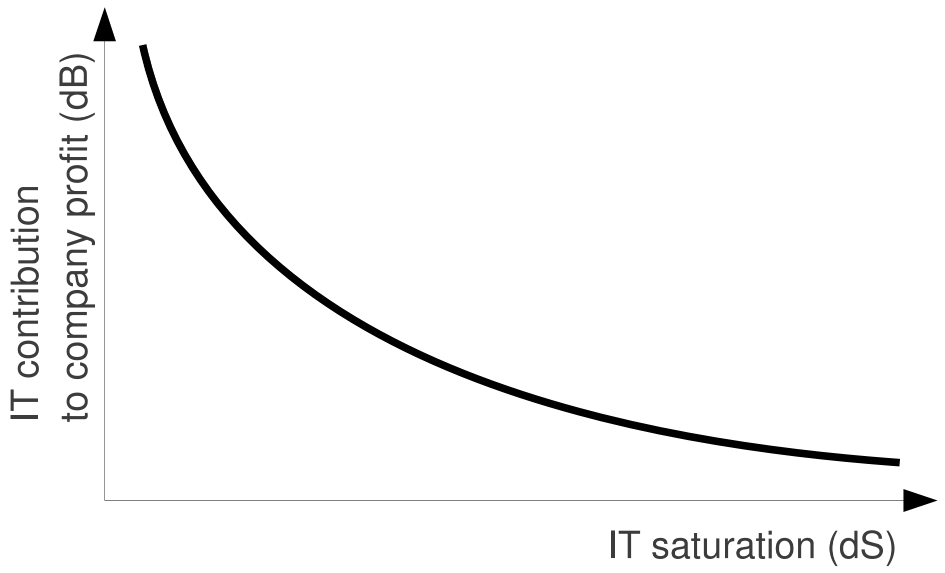


Illustration 1: IT contribution to company profit

Summarising the current situation (Illustration 1), the marginal additional value contribution by IT declines as the degree of automation reaches its maximum[4]:

4 Definition of the term maximum is only relative to what is achievable at a certain point in time. As automation is an ongoing process, there are no 100% of IT saturation (if one accepts the presence of humans in the process chain)

$$\lim\left(\frac{dB}{dS}\right) \to 0$$

with dB being the change in IT contribution to company profit (benefits) and dS the degree of IT saturation. Put simply, Illustration 1 says that any further automation will not generate enough additional benefit to justify its implementation[5].
Further, IT cost tend to increases with higher degrees of automation. At the same time, top management demands reduced IT cost and cuts IT budgets, disregarding demand for innovation, replacement of outdated IT services or required IT-related research and development (Illustration 2).

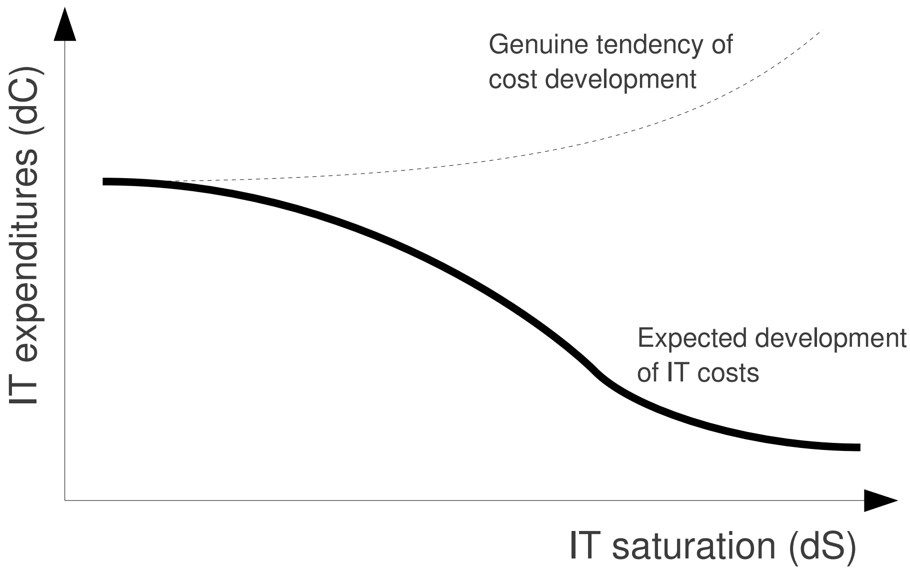


Illustration 2: Expected vs. prognosed development of IT expenditures

$$\text{with } \lim\left(\frac{dC_g}{dS}\right) \to >1 \text{ and } \lim\left(\frac{dC_e}{dS}\right) \to 0$$
$$\Rightarrow \lim\left(\frac{dC_g}{dC_e}\right) \to \infty \wedge \lim\left(dC_g - dC_e\right) \to dC_g$$

with dC_g being the genuine cost development, dC_e the expected cost development and dS the IT saturation level.

Simply put, if IT expenditures are expected to decrease, there will be no financial reserves to invest in innovation. In fact, all additional expenditure into new investment will stall, all financial resources put into maintaining an ageing and outdated IT infrastructure.

This trend is not going to change in the foreseeable future.

5 For new investments, demonstrating a positive business case will be increasingly harder, additional automation might even be economically counterproductive

1.2 Research questions

We have seen that IT budgets are under pressure, both externally and internally. External cost drivers are increased license and maintenance fees as well as higher cost for service providers. Internally cost for personnel, training and non-productive times rise analogue to the increased complexity of IT systems.

This development leaves little to no room for innovation and new development. Without substantial change, IT is bound to end as a provider of commodities without added value for the company. This counters the effort of IT managers and CIO's to provide better services, aspire to be an innovation partner for operational business and gain influence in the management and operation of companies.

Marginal contributions to company profits by IT can only increase if new potentials are tapped. Identification and exploration of new potentials to increase the value contribution of IT is core task of strategic IT management.

Open Source Software (OSS) offers such a potential. OSS saves on license and maintenance fees, operates on a variety of different hard- and software platforms. It offers some strategic advantages reducing operational risk and expenditure. Methodologies applied in the creation and maintenance of OSS have been successfully adopted in different industries (see Enkel E., et.al., 2009 and Chesbrough H.W., 2003 on Open R&D and Open innovation, Kell L.T., et.al, 2007 on implementing OSS to support environmental management as well as Lettl Chr., 2010 on improving LEGO using paradigms of open source). The success described there gives rise to reapplying OSS strategies and methodologies onto IT itself.

This paper intends to answer the following questions:

Q1: What is the value of OSS in strategic IT management?
Besides its traditional application in centralised server systems, this question tries to identify applications of OSS and the methods applied in OSS development and maintenance.

Q2: Which basic preconditions must be met to successfully adopt OSS in strategic IT management?
Promotional and repressive factors to OSS implementation shall be identified. The implication of open standards and communities on OSS will be discussed and qualified.

Q3: What economic benefits can be gained by companies and their environment by using OSS?
This question intends to shed light on primary and secondary potentials OSS offers to

enterprises. In order to measure effective development, a set of metrics shall be developed.

1.3 Methodical approach

Idea
The basic idea for this research was drawn from a company wide re-evaluation program of IT expenditures inside the Austrian Railway Company in 2010. Research was initiated through personal observation and critical discourse with some open source experts in Vienna.

Literature review
As OSS gets increasing public and management attention, there is a wealth of books, articles and essays available that deal with every aspect of OSS. Most literature is available online and freely accessible. This reflects the very nature of OSS itself. Thorough literature review on OSS related to strategic IT management was carried out to lay the foundation for in-depth analysis of OSS benefits with respect to economical and strategic parameters. Findings were critically evaluated and categorized.

Conceptual work
A concept for OSS in strategic IT management including:

- a comprehensive list of strategic benefits using OSS
- a weighted list of prerequisite critical success factors and
- a set of perspectives and parameters for use in IT-Balanced Scorecard monitoring

based on grounded theory concluded the research.

Critical evaluation
Findings and conclusions were critically evaluated against historical and current development in the open source environment. Where necessary to support further arguments of this paper, these historical references have been included in the text.

1.4 Research proceedings

Screening
A first review of available literature was based on search runs using the Google internet search engine (http://www.google.com) with the following list of keywords:

> *open source, open source software, strategic management, strategic IT management, IT management with open source, open source development, limux, wienux, open source projects, OSS*

This search was extended using Google Scholar Beta search engine (https://scholar.google.com) and Google Books search engine (http://books.google.com).

Additionally, similar internet searches were carried out using Wikipedia (http://www.wikipedia.org), Amazon Germany (http://amazon.de) as well as the library catalogue of the University of Applied Science FH Technikum Wien (http://aleph18-prod-sh2.obvsg.at).

Finally TU Berlin annually publishes an open source compendium, the "*Open Source Jahrbuch*" (http://www.opensourcejahrbuch.de) which provides a compendium of research papers on open source topics. This concluded the primary search for relevant sources.

A long-list of information material identified as being relevant with respect to open source in combination with strategic IT management was established and used as input for literature research.

All material in this long-list is available online and on the CD accompanying this thesis.

Isolating scope of research
Based on the research questions in chapter 1.2 the material in the long list was reviewed in detail with respect to challenging the research questions, its relevance to the subject rated, basic concepts identified, solutions and extended research questions from previous material extracted and formulated.

Only selected essays from the "*Handbook of Research on Open Source Software*" (St.Amant K., Still B., 2007) and the "*Open Source Jahrbuch*"-series (Gehring R.A., Lutterbeck B., 2004 - 2008) were considered it this research. Contributions dealing with the history of OSS, detailed implementation reports on specific applications or discussion on ethical issues were not considered suitable to the research at hand.

The short-list of contributions that this thesis is based upon is referenced in the Bibliography at the end of this paper.

Research status and critical evaluation
The current status of research was asserted. Recent years brought considerable changes to the OSS related IT industry. Some of the material evaluated is based upon outdated information. MySQL for example was referenced in several studies as an open source software provider. MySQL was acquired by Sun Microsystem Inc. Similar material can be found on OpenOffice.org which was staffed and financially supported by

Sun. Oracle purchased Sun in 2009 getting a selection of OSS[6] in a single transaction.

While this was a setback to the open source community, results of these respective studies remain significant.

As for the cases of MySQL and OpenOffice, the open source community reacted in a novel way: MySQL was forked to MariaDB, SkySQL and Drizzle, OpenOffice forked to LibreOffice. So, in principal, both OSS environments live on under different names.

Chapter 2 summarises what was found and considered relevant to answer the research questions stated in chapter 1.2.

Concept development
Based on the material evaluated in chapter 2 a concept was developed that gives a comprehensive list of strategic benefits OSS can contribute to strategic IT management. A table of strategic IT management tasks was created, OSS benefits related to these tasks.

These contributions were evaluated and a list of critical success factors defined. These factors were rated and ranked.

Finally these critical success factors (CSF) were modelled into perspectives and key performance indicators (KPI) that can be used in IT balanced scorecards to monitor, evaluate and optimise OSS performance.

1.5 Assumptions, requirements and preconditions

This paper is based on some assumptions.

The primary assumption is that there is strategic and economic benefit in using OSS which neither have been acknowledged on a broad scale nor have been realised yet.

Literature sometimes rate OSS as being superior to proprietary software (PS) with respect to stability, innovation and dynamic adjustment to changing requirements. Pro PS literature emphasises lack of support, non-committal road maps and the missing responsibility by developers due to the voluntary nature of software development in OSS. Both sides are right in their own way. As literature about successful or failed projects shows, there are some differences in planning and production between OSS and PS but these are no show stoppers to provide successful software or solutions.

6 Sun Microsystems Inc. was owner and provider of Solaris, Java and Glassfish, which was made available as open source. Further Sun owned MySQL and StarOffice and supported the development of OpenOffice.

This paper therefore does not assume either OSS or PS being superior. It strictly tries to follows the path of economical and strategical corporate value.

The thesis builds on a very brief introduction into strategic IT management. It is assumed that the reader has a general idea about Gälweilers (Gälweiler A., Malik F., 2005) model of strategic management and control parameters. An online version is available at Google books (https://books.google.com, search term "*gälweiler malik*").

Some of the material was taken from German language sources. Where appropriate and feasible the material was translated into English. In all other cases, German words and phrases are described in the text.

The reader is not required to having used or being acquainted to OSS. It is not impedimental though.

The concepts developed in chapter 3 are concerned with internal IT aspects of enterprises and companies who's primary business purpose is to operate and use IT services and products. It is not specifically targeted towards companies providing IT services as their primary revenue stream. In those cases, the business models suggested had to be chosen differently.

Throughout this document some abbreviations are used. Their interpretation is summarised at the end of this paper (page 100).

1.6 Expected results

This paper shall provide the following results:

- a comprehensive list of strategic benefits using OSS
- a weighted list of prerequisite critical success factors
- a set of perspectives and parameters for use in IT-Balanced Scorecard monitoring

As a side effect it shall indicate areas of operational implementation.

1.7 Document structure

Chapter 1 briefly describes the environment and situation, IT departments and CIO's face nowadays. It defines the scope of research carried out in this paper, formulates research questions and describes the procedure towards the final results.

Chapter 2 reviews and critically evaluates current literature with respect to strategic IT management, open source software (OSS), use cases of OSS migration projects and metrics for performance measurement.

Chapter 3 provides original conclusions, deductions and suggestions and resembles the added value this thesis provides.

Not all considerations can be taken without contradiction and challenging. *Chapter 4* critically evaluates the results from chapter 3 and suggests further research.

Chapter 5 summarises the results and suggestions of the paper by highlighting selected, non-trivial and innovative findings. Comprehensive discussion and details from chapter 3 are cross-referenced.

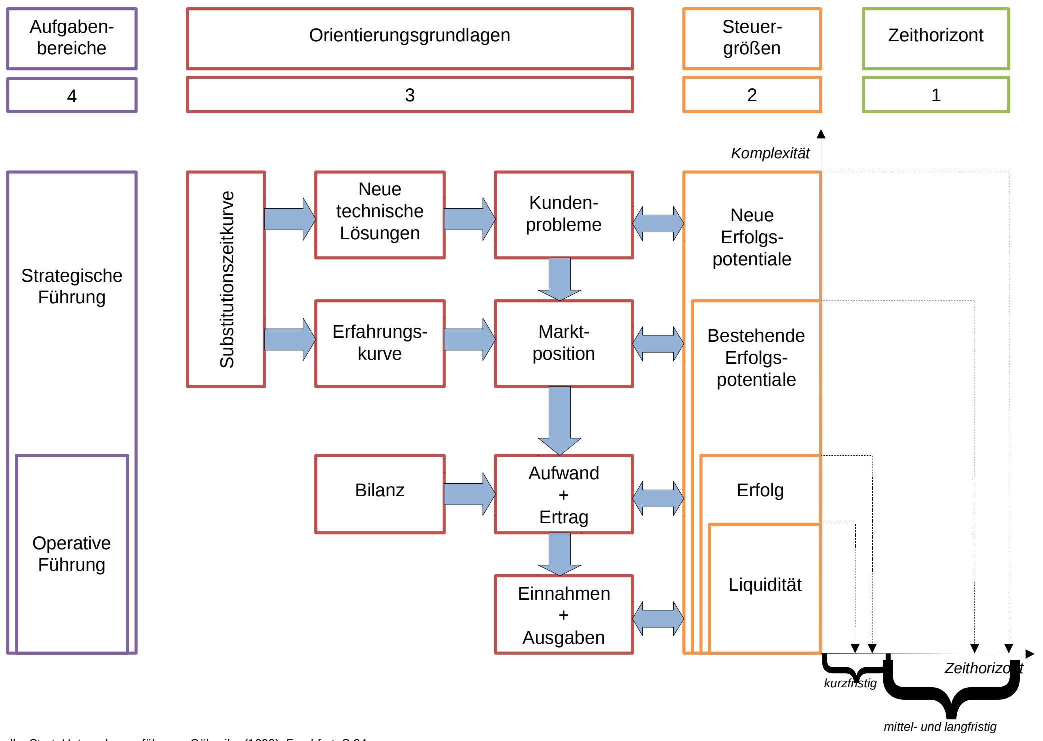
Value contribution of this thesis to the subject is denoted by grey shading

2 Research status

This chapter will describe and summarise the current status of research and the material and information available covering strategic IT management in combination with OSS.

2.1 Strategic IT management

Gälweiler A., Malik F., 2005 define strategic management as the sum of activities that preserve existing and create new (future) potentials for corporate success:



Quelle: Strat. Unternehmensführung, Gälweiler (1990), Frankfurt, S.34

Illustration 3: A.Gälweiler: Strategic management and controlling parameters

„Demgegenüber besteht die Aufgabe der strategischen Unternehmensführung darin, so früh wie möglich und so früh wie notwendig für die Schaffung und Erhaltung der besten Voraussetzungen für anhaltende und weit in die Zukunft reichende Erfolgsmöglichkeiten, das heißt "Erfolgspotentiale" zu sorgen. Das Erfolgspotential ist die bei der strategischen Unternehmensführung im Mittelpunkt stehende Führungs- bzw. Steuerungsgröße.“

(Gälweiler A., Malik F., 2005)

(Opposed to that (operational management) strategic management has to establish and maintain best preconditions for sustained success potentials

that reach far into the future. These potentials are parameters for management and controlling)(translation by the author)

Illustration 3 shows the interdependence between operational and strategic management, time frames (“*Zeithorizont*”) for both management domains as well as areas of significance (“*Orientierungsgrundlagen*”).

In order to create strategic success potentials, strategic management follows an overall plan, is detached from specific time frames, comprehensively considers all influential parameters and consists of coherent steps. All this shall guarantee future success.

Gälweiler acknowledges that operational goals are easier to define, explain and follow. Thus they are a constant threat to strategic goals and decisions but are capable only to provide suboptimal, strategically inferior results.

Tiemeyer transfers this model into strategic IT management. According to his approach, the core question of strategic IT management is: “*Are we doing the right thing?*” as opposed to (operational) IT management, which asks: “*Are we doing things right?*”. He also states that strategic IT management embraces and extends on operational management (Tiemeyer, E. 2010). Illustration 4 demonstrates the transferred model.

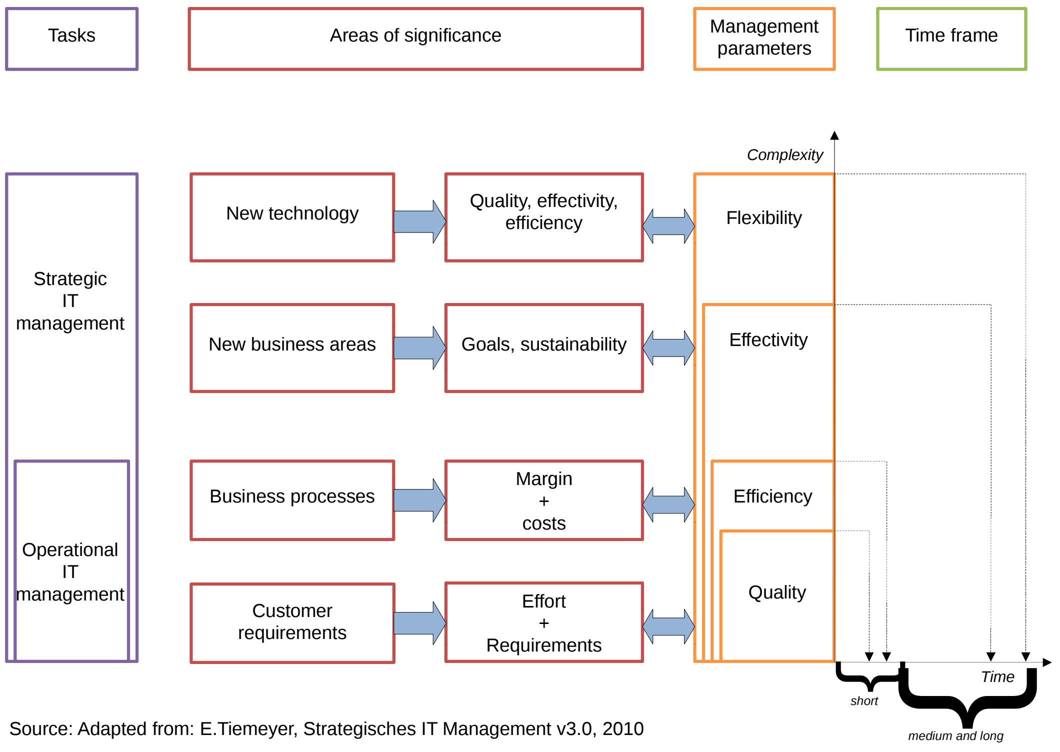


Illustration 4: E.Tiemeyer: Model for strategic IT management

Consequently, typical tasks of strategic IT management are:

- strategic planning of IT systems landscape
- implementing and managing IT architecture
- sourcing management (outsourcing, recruiting, partner management)
- management of IT program- and project portfolio
- IT value management and IT governance
- IT lobbying (Tiemeyer: *organisational positioning of IT inside the company*)

This is broken down into:

Strategic goals: • ROI, economic value • securing future development • customer satisfaction • optimized processes • empowered personnel • secure IT systems • availability	Strategic activities: • IT Vision and Mission • develop IT strategy • plan IT architecture • prepare sourcing decisions • IT lobbying
Typical roles: • mediator between top management and users • plan and communicate targets • program- / project management • escalation management	Typical situations: • high management – employee ratio[7] • change management • ambitious stakeholders (and management of their expectations) • resolution of conflicting goals • highly complex IT systems • environmental demands ("Green IT"[8])

Table 1: Strategic IT Management: Goals, Activities, Roles, Situations

7,8

Tiemeyer identifies 5 capital sins committable during the process of implementing strategic IT management. They describe essential (but missing) preconditions (existence of a business vision, definition of the role of IT) as well as wrong implementation of the strategic process. Here is the list of capital sins in strategic IT management as identified and defined by Tiemeyer, E. 2010:

7 High management – employee ratio: The mathematical value actually is low as there are less managers managing more employees

8 Green IT: While Green IT is a keyword summarising a lot of different goals, cost cutting of energy expenses is often considered to be important

Missing systematic process	If there is no predefined process to develop IT strategy, the results will be arbitrary
Product standards as starting point	Products should be selected in accordance with IT strategies, not vice versa
Missing business vision	Missing correlation between business and IT strategies. IT will miss the market
Diverging understanding of the role of IT	Without support and commitment by top management, the role of IT will diminish
Project “IT Strategy”	Development of IT strategy is a process rather than a project with a predefined end

Table 2: 5 capital sins of strategic IT management

In order to successfully establish strategic IT management, the following additional necessary preconditions must be met. Existence of these preconditions must be considered to be critical success factors (CSF's) to strategic IT management:

- strategic alignment of IT with the overall business strategy
- IT architecture, IT infrastructure, IT processes, IT organisation (is IT applied effectively and efficiently)
- IT leadership
- IT governance
- Value contribution of IT to the business

Inge Hanschke takes a different approach to strategic IT management. She asks two questions:

What part does your IT play in the enterprise?

and

What is the current performance potential of IT?

(Hanschke I., 2010)

She offers four different models similar to Forrester's pyramid:

- IT as a cost factor (provider of commodity, necessary evil)
- IT as an asset (delivers excellent quality and efficiency)
- IT as a business partner (contributes to value propositions and enterprise strategy)
- IT as an enabler (shaping new business models)

The four types are considered stairs with increasing value to customers, users and the enterprise. The first two steps require operational management, while the last two require strategic management.

For the second question about current performance potential of IT, again four alternatives are given:

- ensuring business operations stay up and running (operate IT reliably and securely)
- appropriate, cost-effective IT support (competing for cost leadership with external providers)
- securing the future viability of the IT landscape (establish technological standards and frame evolutionary development of IT landscape)
- optimising and enabling the business (questions potential for IT innovations)

From there she infers future strategic positioning of IT ("*Where Do We Want to Go*" or "*How Does IT Wish to Position Itself in the Future*"). The options offered provide 7 different strategic directions an IT department can head to:

- stand out from the competition through individualisation or cost leadership
- be faster than competitors
- create and optimise access channels for customers (easy access to IT)
- make customers more loyal – and more dependent
- efficient assignment of resources
- cost savings
- agility to change the business model

Combinations of options are possible and depend on the actual situation of the evaluated IT. Building on the results of this evaluation Hanschke deduces strategic objectives, IT goals and strategies. Her methodology does not explicitly state CSF's but it offers some principles for strategic guidance (see Table 3).

Some strategies specific to isolated subjects like technical standards, innovation, investment, sourcing and vendors are offered as well a methods to analyse and visualise strategic IT portfolios. Her book continues to describe a methodology of strategic IT architecture management to the controlling of the strategic objectives and IT portfolio.

While not all of the previous strategic directions are disjunct, some remain relevant. These are:

- cost
- speed
- access and
- customer loyalty

Selecting software solutions	• Best of breed • Make or buy
Appraising projects	• Prioritise core business • Infrastructure projects first
Design principles	• Avoid heterogeneity • Technical structure follows business structuring • Avoid redundancy • Only lead systems modify master data • Single point of distribution of master data
Procedures in strategic IT planning	• Divide & conquer • Tuning (optimise and stabilise) • Housekeeping (opportunistic development) • One IT (regards post merger situations) • Tried & tested over new • Replacement strategy (concerns legacy systems)
New applications	• Big bang • Evolutionary approach
Increasing technical quality	• Flexibility • Deconvolution (reduce system interconnections) • Decoupling • Encapsulation
Efficiency	• Virtualisation

Table 3: Principles for strategic guidance

We have seen how strategic IT management was derived from strategic management in order to maximise future potentials and successful contribution of IT to company profitability. There are differences in the approach and methodology used but the goal remains the same: build up potentials.

In the following chapter we will investigate potential strategic benefits, OSS can provide and contribute to the enterprise.

2.2 Strategic benefits of Open Source Software

Open source software (OSS) has been described in detail in several places (St.Amant K., Still B., 2007, von Krogh, G., Spaeth S., 2007, Gehring R.A., Lutterbeck B., 2004-

2008, The open source initiative: http://www.opensource.org). We will refer to previous work only as far as is required to maintain our argument and answer the research questions from chapter 1.2.

According to the OSI definition (http://www.opensource.org), OSS must comply with the following criteria (taken from the website):

1. *Free redistribution (by anyone, in any combination, possibly free of charge)*
2. *Source code (included)*
3. *Derived works (must be allowed under the same license terms)*
4. *Integrity of The Author's Source Code (allowing patched-only or renamed successive versions)*
5. *No Discrimination Against Persons or Groups*
6. *No Discrimination Against Fields of Endeavour*
7. *Distribution of License (no requirement for additional licenses)*
8. *License Must Not Be Specific to a Product (neither in combination nor in isolated use)*
9. *License Must Not Restrict Other Software (no exclusion of any sorts)*
10. *License Must Be Technology-Neutral (not requiring any prerequisital soft/hardware)*

Some of these criteria are hard to achieve, hard to apply and hard to interpret. Freedom to redistribute includes both the freedom to do it for free as well as to charge for extra services. The inclusion and availability of source code gives rise to debates concerning quality and security issues. Non-discrimination against people and implementation includes citizens from third world developing countries as well as terrorists. Unrestricted neutrality towards other technology sometimes conflicts with existing domestic law[9]. (St.Amant K., Still B., 2007).

In order to isolate strategic benefits OSS offers, we first conduct a SWOT analysis. Erlich Z. and Aviv R. in Chapter XV of the “Handbook of Research on Open Source Software” provide a categorized overview (Table 4: OSS Strength and Weaknesses).

9 In certain countries use of software libraries that circumvent media protection schemes are illegal. Similarly, free distribution of patented audio and video codes like the Fraunhofer MP3 codec is prohibited.

Strengths	**Weaknesses**
Freedom of use: • Free access to software and source code • independence from single vendors • platform independence • free upgrade at users own pace Evolution of software: • voluntary developers • quick bug fixes • code reuse Time, cost and effort: • lower development cost • quicker bug fixes • no license fees • flexible maintenance fees • code reuse • no time and budget restrictions during development Quality of software: • reduced number of bugs • user feedback • constant peer review • intrinsic code quality (out of programmers self esteem) • vulnerabilities found quicker • alternative code distribution channels Advantages to companies and programmers: • efficient use of knowledge • learning by example • gaining programming experience • opportunities to collaborate • rapid development	Management: • challenging planning and delivering OS community projects • difficult coordination and collaboration • complex resource allocation and budgeting • fluidity of developers (varying interest) • inadequate existing cost and business models • generating revenue is demanding • good programmers are hired by PS companies Quality and security: • lack of quality documentation • applications not always intuitive • no generally accepted style guides • competition for qualified programmers • open source invites cyber terrorists • fewer applications per problem domain available

Table 4: OSS Strength and Weaknesses

Strengths are typically found in the technical and personal domain while weaknesses of OSS can be found in the area of management. Arguably security and quality issues are mentioned to be both strengths and weaknesses.

Additional literature mentions further strengths and weaknesses. Table 5 summarises them in unsorted order (Hnizdur S., 2003, Wieland in Gehring R.A., Lutterbeck B., 2004-2008, Lutz B. et.al., 2004, Reifenstein J, et.al, 2004, St.Amant K., Still B., 2007):

Strengths	**Weaknesses**
Compatibility: • Suitable for certain (not further specified) types of projects Availability: • Easy access to OSS Economical: • demand driven maintenance fees • reduced effort to handle license agreements • beneficial to the local economy • long term savings • license model allow flexible development • low TCO Community: • community carries most of the development cost	Quality (amended): • Lack of usability Legal issues: • legal uncertainties and risk through software patents • warranties and guarantees Compatibility: • limited network effects (no broad installed base) • limited support for proprietary document standards Adoption: • difficult migration path • slow adoption by companies Economical: • OSS model open to free rider syndrome[10]

Table 5: Additional strengths and weaknesses (categories added by author)

10

While Erlich and Aviv provide a comprehensive list of strengths, their compendium ignores such important issues (threats) as software patents, legal uncertainties and the risk of large vendors using their patents to raise TCO for OSS[11] or push the OSS solution out of market altogether.

Another argument in favour of proprietary software is lack of vendor warranty with OSS. Regarding warranty as a type of risk management helps finding solutions for OSS. Risk can be

- transferred (via software insurance)
- prevented (via extended quality management)
- mitigated (via backup solutions) or
- accepted

10 Free rider: Profit oriented companies that take OSS, modify and repackage it and sell the resulting product under a different license

11 By either forcing companies to build a reserve or paying for a license

All four approaches must be provisioned for in business cases for OSS projects by allocating financial resources to risk prevention.

As can be seen in previous migration projects like LiMux (Munich, Stuckenberg B., 2007), Wienux (Vienna, MA14, 2009), Treuchtlingen, Schwäbisch Hall or NIVADIS (Niedersachsen, Stuckenberg B., 2007), compatibility through open document standards and a clear migration path are issues that have been underestimated during project preparation. They had been addressed on the fly during migration projects[12].

OSS in the past has been used to solve problems in specific problem domains. The prominent LAMP stack (Linux, Apache, MySQL, PHP) is a typical example where the solution has grown with increasing demands. Some problem domains are dominated by OSS, having virtually suppressed the development of commercial software. CMS (Content management systems) are a well known example where OSS has suppressed major commercial development.

Opportunities	Threats
Business opportunities: • Strategic value of OSS indicated • Adoption by large IT vendors (IBM, HP, Oracle, ...) • Proven business models available • OSS acknowledged as business alternative Adoption: • Large scale adoption in 3rd world and developing countries • OSS being complementary to sold products • Wide adoption by hardware and embedded system vendors • OSS methods adopted in different industrial areas (open innovation) Awareness: • Publicity and awareness	Legal issues: • Changing legal regulations • Patent laws in the EU • Patent vaults • Monopolised markets Community disintegration: • Outdated, unchallenging projects • Eliminated personal benefit • Members outgrow group Technology: • Heterogeneous software pool • License issues • Specific application domains not covered by OSS Economical: • Migration projects infeasible • Reference migrations unsuccessful

Table 6: OSS opportunities and threats (categories added by author)

12 There are still some unresolved issues. While OpenOffice can read proprietary Microsoft file formats with sufficient accuracy, transferring files into Microsoft Office is not that easy (Gehring R.A., Lutterbeck B., 2004, Renner Th., et.al., 2005). This causes compatibility issues when exchanging data with external communication partners. A solutions regularly offered is the use and exchange of portable document file formats (PDF)

Let us now examine opportunities and threats. The "*Handbook of Research on Open Source Software*" again is a good source of reference. Additionally, Reifenstein J, et.al, 2004, Lutz B. et.al., 2004 and the "*Open Source Jahrebuch*" series provide a wealth of input on the matter (Table 6). Categorisation of Table 6 was added in this paper to facilitate further analysis on the subject. Refer to Appendix A for a complete overview of the SWOT analysis.

In order to derive suitable strategies utilising OSS, the SWOT matrix is transformed into a TOWS matrix[13]. The method suggests four derived strategic directions. They concern the four main fields in the matrix:

- SO: Utilize to best efforts
- ST: Recover and restore strength
- WO: Control and monitor competition
- WT: Rescue and survive

General strategic directions have been colour coded for better reference. This should give a first indication of where OSS offers strategic benefits or requires additional resources to mitigate implementation risk.

TOWS Matrix		O			T			
		Business opportunities	Adoption	Awareness	Legal issues	Community	Technology	Economical
	Freedom of use							
	Evolution of software							
	Time, cost and effort							
	Quality							
S	Advantages to programmers and companies	Utilize to best efforts			Recover and restore strength			
	Compatibility							
	Availability							
	Economical							
	Community							
	Management							
	Quality and security							
	Legal issues							
W	Compatibility	Control and monitor competition			Rescue and survive			
	Adoption							
	Economical							

Table 7: TOWS matrix derived from SWOT analysis

13 For further reference see: http://www.mindtools.com/pages/article/newSTR_89.htm

Following is a brief discussion of strategic benefits that are anticipated both in academic literature as well as enterprise expectations. Where appropriate, these benefits will be cross-checked with current development in the open source IT industry. The following chapter will discuss important strategic benefits.

2.2.1 Vendor independence

Proprietary software offers high levels of network effects[14]. To maintain the benefit, users must follow the development of vendors versioning policies. Negligence to do so will lead to increasing data and communication incompatibilities.

Commercial software vendors provide filters for import and export of data. If the ratio of import to export filters favours import filters, lock-in effects take effect. If the afore mentioned ratio reaches its maximum, users of this software have no way to migrate to alternative software solutions without loosing pre-recorded information.

$$\text{with } L=\frac{\sum F_{imp}}{\sum F_{\exp}} \quad \text{if } \lim L \to \infty \quad \text{then } C_M = C_{nS} + C_{de} + C_{tl}$$

with L being the Lock-in[15] effect, F being filters for import and export respectively. C_M being total cost of migration, C_{nS} is the cost of the new software system, C_{de} the cost for data copying and data entry and C_{tl} being transitional cost.

As C_{de} and C_{tl} grow exponentially with the use of software systems, this demonstrates that once customers are locked into a specific software system, migration to alternative systems might not be feasible. Vendors that have achieved this level of control can then increase license and maintenance fees just below the total cost of migration[16]. As C_M continues to rise, so do license and maintenance fees, because the situation never tilts. Finally, IT budgets get absorbed by maintenance of legacy systems.

To prevent this scenario at an early stage, in developing countries like South America or the Far East independence from dominating software vendors ranks high in the list of strategic benefits (Chapter VII ff., St.Amant K., Still B., 2007, Gehring R.A., Lutterbeck B., 2004-2008). Independence from specific vendors has been a driving factor for the

14 *"In economics and business, a network effect (also called network externality or demand-side economies of scale) is the effect that one user of a good or service has on the value of that product to other people. When network effect is present, the value of a product or service increases as more people use it."* (http://en.wikipedia.org/wiki/Network_effect)

15 There are other factors adding to lock-in effects. They are not considered in this formula

16 Microsoft Outlook offers many import filters but only a rudimentary set of export filters. These export filters do not even cover all items stored in the internal data structures of Outlook and the corresponding Exchange server. Thus, migrating to a different system can be achieved only by accepting some data loss.

sustained efforts in the LiMux project (Stuckenberg B., 2007).

To achieve vendor independence, IT systems need to be open to inspection and review, use open and widely acknowledged document and communication standards[17] and facilitate a wide range of export filters (preferably: $\sum F_{imp} \leqslant \sum F_{\exp}$).

While open source currently does not fulfil the last requirement, generally OSS supports **open, well documented standards** and provides enough **filters** to prevent solution and vendor lock-in.

2.2.2 Availability of source code

Source code is the foundation of software development. It describes the solution to a certain problem in the most accurate and unambiguous way.

Source code is required
- to regenerate (recompile) working software
- inspect the inner algorithms in case of errors and malfunction
- extend existing software, in case of additional business requirements
- modify internal data structures and interfaces in case of extended requirements to store and manipulate data
- discover potentially negative side effects (from simple software failures like buffer overflows, potential code injection to established malicious code)
- to prevent monopolised software development or
- to prevent disruption of business continuity due to vendors going out of business

While there is common agreement that only few people will ever look into source code, the opportunity to do so (even by third party contractors) is considered a strategic benefit (Reifenstein J, et.al, 2004, Gehring R.A., Lutterbeck B., 2004-2008).

Availability of source code also **reduces the risk** for companies in case a software vendor goes out of business or monopolises the market. This has been demonstrated in two cases recently:
- MySQL was eventually purchased by Oracle (after a short term serving with Sun). Oracle has an arguable understanding of what OSS and the open source community constitutes. The founder of MySQL branched the source code into MariaDB. MariaDB is intended to substitute MySQL for open source projects.
- OpenOffice.org having been developed under auspices of Sun was part of the Sun-Oracle deal as well. As with MySQL, active members of the developer community initially tried to implement urgently necessary adaptions to the code. Oracle declined support and requested ownership of community enhancements

17 RFC's are such standards (http://www.rfc-editor.org/), see chapter 2.2.3 for more on standards

to be transferred to the company. This led to the branching of OpenOffice into LibreOffice. Prominent suppliers of Linux distributions like Novell Inc. (SuSE[18]) and Canonical Inc. (Ubuntu) have adopted of the newly branched software.

During the Munich migration project, the project team was required to transfer existing business process support functions based on proprietary macros into the OSS environment. Instead of migrating each single macro, a new set of support tools was developed on top of OpenOffice (WollMux: http://www.muenchen.de/Rathaus/dir/limux/wollmux/228227/index.html) extending existing source code. Without the availability of the source code this would not have been possible. Another benefit was achieved by issuing the resulting code to the open source up-stream. Now, the code is partly maintained by the community. This will be dealt in detail in chapter 3.

Recently there have been allegations about backdoors in operating systems introduced by American government organisations (G.Perry, 2010, http://bsd.slashdot.org/story/10/12/15/004235/FBI-Alleged-To-Have-Backdoored-OpenBSDs-IPSEC-Stack, R.Schaefer, 2009, http://www.nsa.gov/public_info/speeches_testimonies/17nov09_schaeffer.shtml). The OSS community reviewed the code and within weeks found nothing that would support such allegations. As for the Microsoft Windows 7 operating system, except for some disclaimers by Microsoft and NSA, there is no such expertise available for Windows 7[19]. This follows a prior incident in which Microsoft and NSA bypassed **security** features in Windows 2000 security subsystems (Advapi.dll).

An aspect of source code availability not duly honoured is the fact, that amendments to the code added by companies can be fed into **up-stream code development**. This requires some effort. However, feeding code up-stream assure that code gets maintained and enhanced by the community and is available in future releases. This alleviates the company from re-iteratively implementing previous code changes to newer versions of core code[20]. It also secures and preserves investment in code development for the company as well as third parties.

Source code can be made available **publishing** it **on the Internet**. There are plenty of source code repositories out (SourceForge, GitHub, FreshMeat, Launchpad to name just a few). Another way to publish source code is to print and publish it as a book. Early

18 After Novell being purchased by Attachmate Inc. it remains to be seen how SuSE will continue its course

19 This does not imply there are backdoors installed in Windows 7. It just states the fact that there is no way for the public to prove these disclaimers

20 Maintaining compatibility between different modules of different versions is a time-consuming task. In the case of Vienna's migration, this effort could not be invested and thus the project stalled

versions of PGP were distributed this way in order to circumvent export restrictions enacted by the US government[21].

2.2.3 Interoperability

Taken in a technical sense, interoperability[22] describes the capability of software to co-operate and work together with other software systems. While sufficient as a starting point for discussion, in order to identify strategic benefits, the term must be understood in its broader sense. Some important issues related to interoperability are:

- provision of standardised and unified interfaces
- standardised, open and well documented data and file formats both for internal storage as well as for data exchange
- established, proven and standardised communication channels and protocols
- language and culture independent semantics of data and information
- standardised user interaction

The demand for technical interoperability is intuitively understood. The sum of bidirectional communication relations C with respect to the number n of available information systems is:

$$C=\frac{n\times(n-1)}{2} \quad \text{with} \quad n=n_s\cdot n_i\cdot n_f\cdot n_p$$

n_s being the number of systems, n_i the number of interfaces, n_f the number of data exchange formats and n_p the number of possible communication protocols. Other factors like time and policy constraints as well as redundancies for improved security are not counted for in this calculation. This complexity has to be integrated in software systems. With every subsystem introduced, the risk of error and failure increases.

Supporting **open standards** secures potential interaction with systems built in the future (as n_p, n_i and n_f tend towards a minimum).

A comprehensive collection of technical standards can be found in the RFC's (http://www.rfc-editor.org). At the time of writing there are 6.129 different RFC's available[23]. Other standardisation organisations are ISO (http://www.iso.org), ECMA (http://www.ecma-international.org/, JavaScript, OOXML and OpenXPS), OASIS (http://www.oasis-open.org/, ODF) or W3C (http://www.w3.org/, HTML, CSS, WS-i,

21 PGP: Pretty good privacy. Export restrictions (http://www.pgp.com/products/export_compliance.html)

22 *“Interoperability is a property referring to the ability of diverse systems and organizations to work together (inter-operate). The term is often used in a technical systems engineering sense, or alternatively in a broad sense, taking into account social, political, and organizational factors that impact system to system performance”* (http://en.wikipedia.org/wiki/Interoperability)

23 Some of those 6129 RFC's are obsolete, some are amendments to previous RFC's

WAI) to name a few. Adoption and adherence to standards will **facilitate future access** to data produced in the past. With proprietary data formats, access to older data requires specific import filters that may not be available any more[24].

As OSS is only gaining momentum, the number of file format filters is still limited. While there is an initiative to support varying hardware (Linux kernel driver project, Hartman G.K., http://linuxdriverproject.org/foswiki/bin/view/Main/WebHome), support for proprietary document formats is in its infancy (Jan Suhr, Gehring R.A., et.al., 2008). While still being an area that needs improvement, **OSS support for interoperability** is a strategic benefit all the same.

2.2.4 Efficient use of resources

Efficient use of resources is a strategic benefit that consists of the following singular items:

- small technological footprint
- code reuse
- development resources taken from OSS community, academic research and revenue driven enterprises
- education and information exchange mainly over fast, inexpensive internet connections
- lower entry cost (due to no or limited purchase license fees)
- lower operational expenses (due to demand-based maintenance fees)

Most of these items are covered in literature on OSS individually.

While there is sufficient information about proposed business cases and migration cost, there is little information about the total cost of ownership (TCO) of OSS. With PS, the proposed period of usage for a specific version is approximately three to five years. Within this period larger or more complex migration projects to OSS do not prove a positive business case. However, **OSS tends to be more stable in terms of application life-cycle** and as the examples of MySQL and OpenOffice have shown, they are resilient to changes of the maintainer and owner of code.

OSS runs on a variety of hardware platforms. Requirements to operate even on embedded systems and mobile platforms enforce **minimized usage of hard- and software resources**. Also it is in the interest of hardware vendors that their product is operational. Their interest does not extend to provide state-of-the-art software solutions however. This leads to documented basic source code as a foundation for further development and extended business opportunities[25].

24 Admittedly, this cuts short on issues of long term data storage with respect to securely making data accessible in the next 100 or more years. Discussing this issue would extend the scope of this work

25 Providing OSS solutions to utilise hardware is often referred to as **widget frosting**, which will be dealt

An extended set of documented binary libraries and code examples invites to **reuse code**. Source code can be enhanced or forked providing a new, extended version of a library. The community decides in a democratic selective process which code survives[26].

Where code duplication seems relevant and necessary, this is documented (see Gstreamer and ffmpeg audio and video codec libraries, http://gstreamer.freedesktop.org/ and http://www.ffmpeg.org/ for just one example). As both versions of source code are available, developers that build on top of any of these libraries can base their choice on sound evaluation of the basic technology. Also with different licenses available, choice can make a difference in the revenue model of the OSS vendor (see Heafliger S., et.al., 2007).

Open source software originates and is driven by community efforts (e.g. KDE, Pidgin), academic research (e.g. PostgreSQL, Python) and revenue driven enterprises (e.g. Eclipse, Apache, OpenOffice). Developers employed by companies get paid to develop and maintain OSS code (Henkel J., 2008). This can happen on a voluntary basis both from the developer as well as the company submitting the source code into the community. While there are some limitations to be considered, it is reuse of resources nonetheless.

Lower entry and maintenance cost is demonstrated in the "*Open Source Catalogue 2007*" (von Rotz B., 2007, Illustration 5). von Rotz claims that OSS causes only around 25% of the cost CSS development expenses ad up to.

with later

26 There are some exceptions to this rule. The linux kernel for example allows for a semi-democratic discussion of code that goes into the kernel. Adoption of kernel enhancements are controlled by a limited number of people. A similar process is followed by different Apache projects as well as the KDE desktop environment

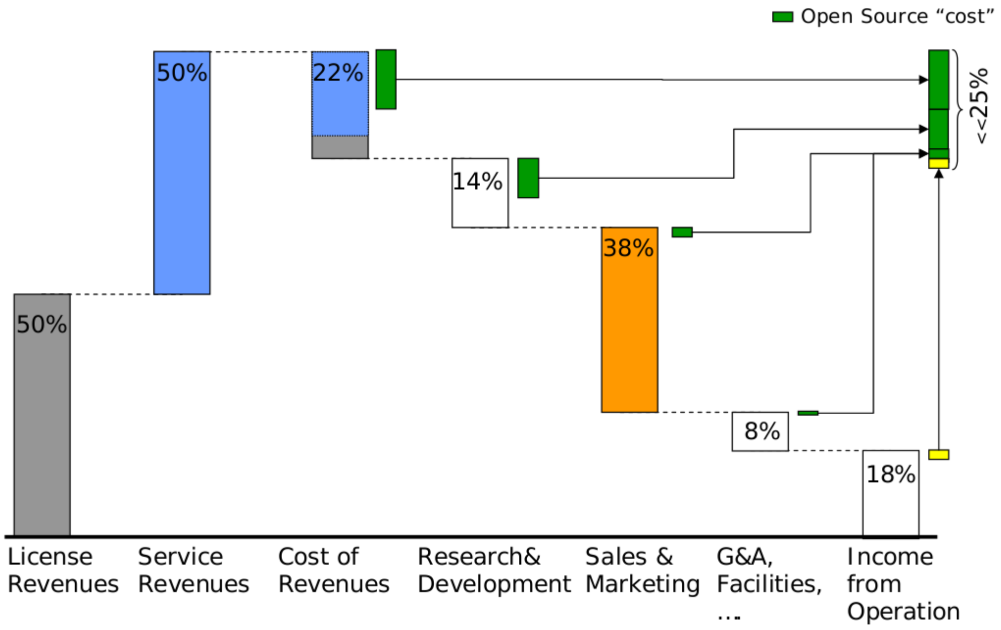


Illustration 5: Cost model comparison: PS vs. OSS (Source: von Rotz B. (2007))

Sempert F., 2009 offers a similar cost model. According to his presentation OSS will save up to 44% on average on comparable expenses (Illustration 6, "Einspaarungen"). Further 25 – 50% of development and implementation cost can be saved by providers of cloud services and SaaS providers.

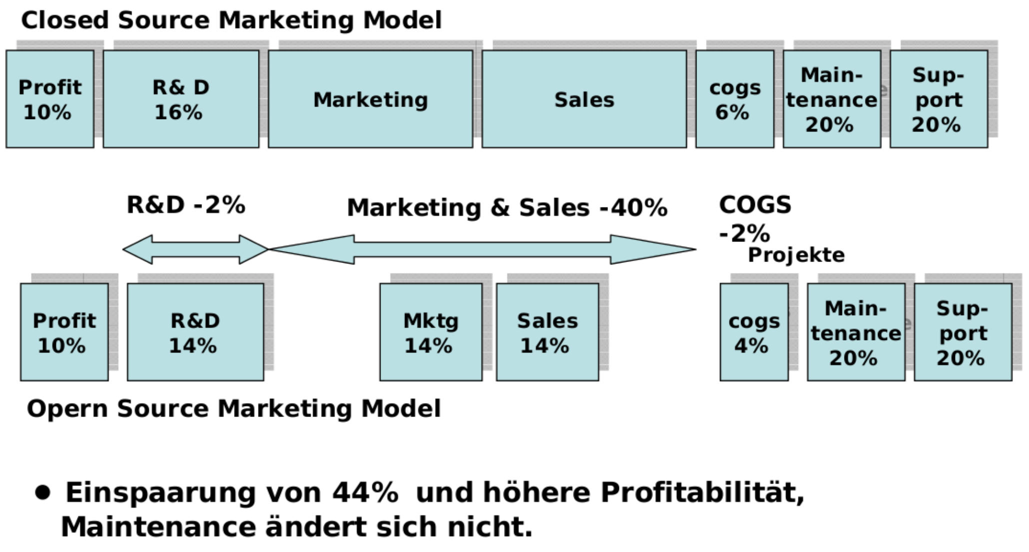


Illustration 6: Cost model comparison II (Source: Semper F. (2009))

As Illustration 6 suggests, Semper does not share von Rotz' optimism regarding cost saving potentials. Even Sempers 44% of cost savings seem arbitrary and cannot be observed without contradiction. Actual cost savings depend on a variety of parameters such as industry branches, software application and operational implementation to name but a few.

Sempert additionally provides a graph demonstrating OSS acceptance over the years in four categories:

- operating systems
- middle-ware and database systems
- open source packaged services (SaaS)
- OSS as components with commercial software

Starting in 2006 acceptance was below 5%. While acceptance for operating systems will reach its peak at 45% in 2012[27], acceptance for OSS as components in mixed environments seems to increase in acceptance beyond 50% with no end in sight. This phenomenon is observable and demonstrates the rising potential of OSS in different applications.

While this picture seems optimistic for future development there are some issues that need addressing. Using **free development resources** is not a sustainable business model. Literature takes free generation of code for granted (Crowston K., et.al, 2007, St.Amant K., Still B., 2007, Gehring R.A., Lutterbeck B., 2004-2008, Lettl Chr., 2010). It is unlikely that software developers will accept the free rides of commercial enterprises for long. The risk of this resource **not being available for the same price in the future** has to be accounted for in the calculation of future business cases.

2.2.5 Other factors mentioned

We have identified four major categories of strategic benefits:

- independence from proprietary software vendors
- availability and accessibility of source code as the foundation for increased quality, security and further development
- interoperability as the foundation to communicate and exchange information
- efficient use of resources as a foundation to provide cost effective systems and services

Literature mentions some more aspects. While valid as an argument in selecting OSS, they lack important properties of strategic factor. They can be viewed as being operational benefits or part of an overall strategic benefit. For completeness they are summarised here:

Technical innovation
OSS sometimes is attributed technological and methodological superiority (Gehring R.A., Lutterbeck B., 2004-2008). This argument lacks deeper foundation. This shall be demonstrated with some examples:

27 As we will demonstrate later, this upper limit of 45% needs to be reflected critically

- Software development: While there are huge numbers of development tools available for OSS development (IDE's, compilers, version management, bug tracking, wikis and download portals), there is little support in the requirement definition phase, deployment or in operation life-cycle management. Proprietary software offers integrated tools to support respective development activities (e.g. Microsoft Visual Studio, IBM Rational Suite)
- Database systems: MySQL offered a database system where the back-end data store is a plug-able module. However, no OSS database system offers data lifetime recording or flashback transactions like Oracle RDBMS 11g does[28]. Currently there are no efforts reported on road maps of OSS RDBMS that will announce this kind of data management
- User experience: KDE offers visually animated user experience with its Plasma library. Mac OS X and Windows 7 have similar visual elements. There is still no OSS driver that handles external displays sufficiently. Windows 7 can handle external screens and beamers intuitively

This list could be extended. There are differences in feature sets. They should rather be attributed to different implementation scenarios than to technological or methodological superiority.

Information and system security
OSS is sometimes considered as being more secure than proprietary software from monopolistic vendors.

Arguably, availability of source code for review helps identify security vulnerabilities. However, code reuse (including already fixed bugs), frequent change of developers (making the same mistakes), as well as the amount of source code available puts this advantage into perspective.

A possible explanation for the anticipated inferiority regarding system security of proprietary software may lie in the fact, that on privately used Windows systems, the primary user has administrative rights per default. Contrary on Linux systems, the primary user has an adjustable set of administrative right that have to be enabled explicitly[29]. A second reason for less vulnerabilities being reported on OSS systems lies in the fact, that there are not so many installations around to make them a promising target for malicious activities. This might change in the future as OSS based IT systems gain wide-spread acceptance.

28 Lifetime recording allows for accessing data as it was available at a specific time in the past. This facilitates audit trails and historical data mining. Flashback transactions allow usage of historical data for analysis and simulation purposes

29 On Ubuntu machines, the first user is added to the sudoer group per default. Sudo however has a restricted set of adjustable access privileges that are enabled only for a certain time-out period

Open Source licenses

Open source licenses are considered a bonus as the owner of the software passes rights to use the software to the licensee. Currently the OSI website (http://www.opensource.org/licenses/index.html) names 67 different licenses, some are more restrictive than others[30]. A detailed analysis of the impact of open source licenses is beyond the scope of this thesis. There are some issues, that indicate that just selecting an OSS license scheme does not cover all legal aspects. Recent judicial discourse in Germany has revealed that the chain of rights transfer must be uninterrupted and conclusive.

The issue of licensing is not limited to OSS. The topic of intellectual property, patents and copyrights also touches the rights of users of closed and proprietary source software. In "*The Penguin has Entered the Building: The Commercialization of Open Source Software Products*", Fosfuri A., et.al. 2007 discuss the impact patents, ownership of intellectual property and trademarks have on the ability and will to issue source code into the open source. Their conclusion is that **companies that own patents are more likely to issue code into open source** than companies that only rely on trademarks or plain copyrights. They are less likely to cannibalise their revenue stream. However, Fosfuri et.al. argue in line that such a donation usually **does not touch core competences** of the company. Code that the company does not want to maintain themselves or wants the testing power of the open source community is submitted into the community.

While ownership of patents might provide strategic benefits (not specific to OSS though) open source licenses are just a vehicle to grant access rights to users. They lack the potentials of strategic benefits.

With the extended acceptance of OSS, the problem of patents, patent pools and collections of intellectual property (IP) will become an obstacle for OSS adoption that needs solution. This issue is beyond the scope of this thesis and should be addressed somewhere else.

Reliability

Reliability of OSS deals with the ability of the software to perform as defined and specified. As OSS generally is not subjected to time-to-market pressure during software development, higher stability and reliability of software might be the result. Where there is pressure to reach the market in combination with OSS usage (e.g. in the automation industries), the general process of quality insurance is more mature than in software development on average.

30 GPL is a flexible and open license. It guarantees openness of subsequent software under the same license. BSD on the other hand allows for commercialising software development and any time

OSS's reliability might be an argument in certain implementations. Some of it might be attributed to availability of source code, some to code reuse. In itself, the argument is not strong enough to be a strategic benefit.

Negotiating power against big software vendors

Reifenstein J, et.al, 2004, St.Amant K., Still B., 2007 and Gehring R.A., Lutterbeck B., 2004-2008 suggest that OSS can be used in negotiations with large software vendors to lower license cost.

Migration from proprietary software to OSS is a long term project including the risk of failure[31] and under-performing project returns. System interconnections have to be broken up during migration, bypath solutions have to be established. There is no guarantee that network effects will remain the same after a successful migration[32].

As large software vendors usually provide professional services to their clients they know about the most complicated IT problems at their customers premisses. Large software vendors usually participate in open source community development in one way or other. Microsoft operates the CodePlex open source platform, provides MS-PL and MS-RL open source licenses and contributes to open source projects like Linux device drivers (http://www.microsoft.com/opensource/default.aspx). Oracle[33] runs Java, Solaris and OpenOffice as open source projects (http://oss.oracle.com/).

After a successful migration, there is little room left for negotiation with large commercial closed source software vendors. Due to the long term nature of OSS projects, there is even little sale potential for added value business to generate as well.

Thus, large vendors have a crisp and clear picture of the pros and cons and therefore the potential success of OSS migration projects. This makes them less receptive to migration threats[34]. None of these arguments supports the theory of strategic potentials being derived from threatening large vendors with open source migrations.

---

31 Currently the ministry of foreign affairs in Germany is reverting its OSS strategy and moving back to proprietary Microsoft systems (http://www.zdnet.de/it_business_hintergrund_die_linux_entscheidung_des_auswaertigen_amtes_meilenstein_oder_randnotiz_story-11000006-41549172-1.htm)

32 In fact it is most probable that reduced network effects have to be compensated for otherwise. This has been demostrated in the Vienna migration where only a small minority used the ODF data format for data exchange and in communication with Vienna's municipal

33 Sometimes with questionable success: http://www.techrepublic.com/blog/opensource/oracle-wins-the-2010-open-source-enemies-prize/2087

34 License costs attribute roughly up to 10% of the total cost of ownership (TCO). Eliminating some or even all of these expenses will not provide significant and sufficient saving potentials.

Support of the local economy
Many commercial software vendors (Microsoft, Apple, Oracle, BMC to name a few) have their head quarters in the United States. With every purchase from these companies, part of the price paid (license and procurement costs), goes to the head office. This drains the local economy from purchasing power and national economic value. With maintenance fees the ratio of exported purchasing power is even higher as the local subsidiaries do not contribute to maintaining or updating software (Reifenstein J, et.al, 2004).

Employing OSS, license fees are either zero or at least a minimum[35]. Maintenance fees for enterprise solutions can either be purchased from the original vendor of the OSS or from local supporting companies. Thus a large fraction of ongoing expenses runs back into local economy.

This argument holds true for small and medium companies employing OSS. It is only partly true for government institutions[36]. During the Munich migration project, local OSS supporting companies were involved in the successful transition (http://www.muenchen.de/Rathaus/dir/limux/89256/index.html). As problems started to accumulate, the project team invited IBM to substitute and step into the breach in order to rescue project success. IBM's revenue mainly go to the U.S.

For large, internationally operating enterprises this argument is irrelevant. Service and support fees will be procured from the distributor or reseller of the software. Usually they charge a small percentage for brokering the software. The major amount of license and service fees goes to the OSS vendor who rarely resides within national boundaries.

While initially being an important operational consideration, this argument does not found strategic benefits any more.

Flexible cost management
Reifenstein J, et.al, 2004 also suggest that OSS allows for dynamic cost distribution and thus more flexible cost management. As the core of OSS is available in source code, additional functionality and amendments to the software can be developed individually and then contributed to the open source again. That way, only the required functionality has to be financed by the company that initially articulates the demand.

35 RedHat and SuSE charge a license fee for their Enterprise editions. Maintenance contracts are offered as well. Canonical charges for their Enterprise cloud servers but mostly provide service contracts.

36 In Europe, governments are forced to call for open tender if they invest in new technology. Initially intended to equal chances between small and large contenders, in IT the effect has been reversed and favours large IT vendors as only they can effort the exhaustive procedures and lower the price enough to win the bid

This has been demonstrated in the Munich migration project. After finding that migrating Microsoft Office macros was too time-consuming and economically infeasible, the project team initiated the development of WollMux carried out by local OSS developers (http://www.muenchen.de/Rathaus/dir/limux/wollmux/228227/index.html). The town of Munich paid for the development and subsequently issued WollMux as an open source project which now is an independent project available at: http://www.wollmux.net/wiki/Hauptseite.

There lies potential in this type of up-stream development and business model. We attribute the main idea to the strategic benefit described in chapter 2.2.4, Efficient use of resources. This idea will be extended further in chapter 3.

Scalable systems

Open source packages are assemblies of many interdependent and interoperational modules. Users are not required to install the full set (if any such can even be defined) of functionality. This makes for efficient use of resources.

Tiny Core Linux (http://www.tinycorelinux.com/) and Damn Small Linux (http://www.damnsmalllinux.org/) are two representatives of Linux distributions with an extremly small demand for resources. They form the basis of embedded system development. On the top of the scale range systems like SuSE Enterprise Server and Enterprise Desktop or Ubuntu Desktop and Server Edition that provide a rich set of enterprise solutions. The website http://distrowatch.com/ offers a wealth of choices regarding open source distributions.

As the Vienna migration project has demonstrated, this **flexibility in scaling installations can back-fire** quite easily. The project team in Vienna decided to use a selection of components and assemble a specific set of functionality: Wienux (http://www.wien.gv.at/ma14/wienux.html). Maintenance of this disparate and disaggregate selection of modules is beyond the scope of IT departments. They cannot effort the human resources to work on a specific distribution like set of OSS as well as maintain ongoing IT operations. Wienux and it's components currently are completely outdated. The user acceptance of the solution decreases and the efforts in provisioning IT resources are infeasible.

The positive aspects of scalability have been acknowledged in 2.2.4, Efficient use of resources. For the rest, this rather seems a risk of using OSS than a strategic benefit. Chapter 2.3.5 details the reasons for the (not publicly admitted) failure of the Vienna migration project.

2.2.6 New potential strategic benefits

Literature offers some use cases that exhibit economic and business development potentials for OSS that have not been explicitly declared as such. In this chapter we discuss these use cases and derive two strategic benefits that were not found in previously available literature.

Adoption of emerging markets for open innovation

Contrary to available statistical data[37], there is a wide adoption of OSS in emerging countries in Africa (Uganda, Tanzania) and South America (Brazil) as well as the far eastern countries like China, Korea and India to name a few (Chapters VII-IX, St.Amant K., Still B., 2007, Gehring R.A., Lutterbeck B., 2004-2008).

Proprietary software was pirated for a long time in these emerging markets, thus access and operation did not provide problems or financial burden. With enhanced copy protection schemes, accessing working copies of PS becomes harder and riskier. Companies in emerging countries are neither willing nor capable to effort the price that legal licenses cost (see university migration project in Uganda).

Governments in these countries fear of being subjected to the influence of U.S. companies (and possibly the US governmental agencies that control them). Brazil has introduced a country wide initiative to use OSS in broad terms. China has put emphasis on OSS in its 5 year economic development plan. As a member of the WTO, China had obliged itself to reduce software piracy. OSS provides an alternative in a fast growing IT market like China's.

Embedded hardware producers use OSS to provide basic functionality with their equipment. Producers in China, South Korea, Malaysia and India retreat to OSS in order to keep license fees low to not to burden their products with extra cost[38].

IT Services (consulting and operations) are provided by a large IT industry in India:

> *“India already accounts for the largest number of IBMers outside of the U.S (it recently surpassed Japan).”*
>
> *(St.Amant K., Still B., 2007)*

37 http://www.w3schools.com/browsers/browsers_os.asp
http://en.wikipedia.org/wiki/Usage_share_of_operating_systems
http://gs.statcounter.com/
The reason for this difference lies in the way data is collected. All of the above three count client access to web sites. The browser used indicates the operating system. This method disregards server systems, browser camouflaging and proxying

38 Embedded linux can be found in products like VCR's, DVD-Players, car control systems, mobile phones and PDA's

In order to provide competitive services, Indian companies have to resort to employing OSS in large scales.

Another emerging issue related to OSS is open innovation and open collaboration. In "*Appropriability, Proximity, Routines and Innovation, How open is innovation?*" Dahlander and Gann lay down their thoughts on systematically tapping into external resources:

> *"Managers tend to overestimate the value of internally developed ideas, and fail to realize the full potential of external ideas."*
>
> *(Dahlander L., Gann D., 2009)*

The basic idea is to share innovation and to multiply usage of innovative ideas into different fields and industries. Sharing innovation inspires new parties to participate and generate new business opportunities. While Dahlander and Gann leave it open how this model generates profit, Enkel E. et.al. suggest that open innovation shall start with R&D (Enkel E., et.al., 2009). They offer a two-step approach to gaining from open innovation:

1. *"**Outside-in process**"*: Knowledge is imported into the company from external resources. Knowledge sources are attributed to[39] clients/customers (78%), suppliers (61%), competitors[40] (49%) and research institutions (21%). 65% were other – not further specified – sources. This process generates revenue through competitive and innovative products.
2. *"**Inside-out process**"*: Bringing ideas to the market in form of sold intellectual properties (IP), co-developed products, multiplied technology and externalised knowledge is another way to achieve open innovation. Revenues are generated by selling IP, licenses[41], multiplying technology and thus generate enhanced network effects in the market.

A special application of the Outside-in process is demonstrated in Lettl Chr., 2010. In his presentation on "*Schöpferische Ergänzung durch Open Innovation: Die Demokratisierung von LEGO*" Lettl demonstrates how LEGO tapped into a creative community to produce new LEGO kits. He even goes so far as to develop a mathematical model on how the activity of an individual community member correlates to commercial success of the resulting product.

Open innovation with the afore mentioned growing markets – while not addressable as a single unit – bear strategic potential for **co-operation in research**, as a **distribution**

39 Duplicate mention allowed in the survey

40 The article leaves open if these 49% should be attributed to industry espionage or licence deals

41 Oracle offers its MySQL database both as OSS under GPL and with a proprietary license that allows further commercialised development

market for new innovation and as production and procurement market for **complementary products** and services to OSS.

OSS as a threat to CSS development

In "*The impact of open source software on the strategic choices of firms developing proprietary software*" Jaisingh J., et.al., 2006 develop a mathematical model that demonstrates that firms produce lower quality closed source software (CSS) in the presence of an OSS alternative, compared to the quality achieved if there is no OSS alternative available. The intuitive reason behind this is that CSS vendors have to pay for their developers. Their incentive to invest into software development directly relates to available market share, achievable prices and possible network effects. With OSS around, all of these parameters shrink.

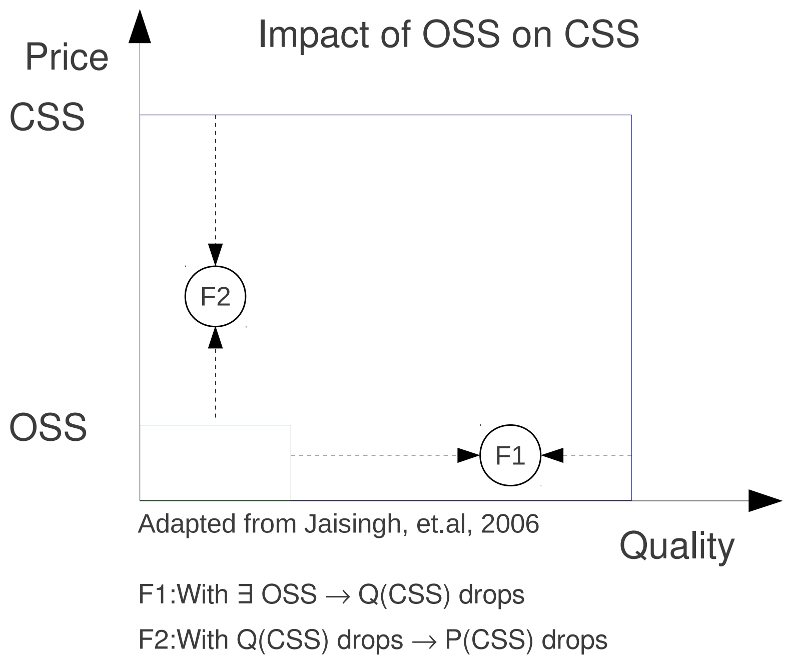
The working paper concludes:

> *"Also since the OSS competes with the CSS for customers, a higher-quality of OSS leads to lower demands for the CSS"*
>
> *(Jaisingh J., et.al., 2006)*

With an OSS alternative in the market, resources developing CSS are reduced (as their invocation is infeasible). In the long run, increasing quality of OSS will lower the quality of CSS. This will lead to elimination of CSS as customers have no incentive to pay for inferior software that can be substituted by high quality software that is available for free.

The model demonstrates quality effects in the presence of 2 CSS (which compete for quality), presence of CSS and OSS (which compete for resources and market shares) and a mixture of both[42]. Illustration 7 presents the findings in a simplified form.

42 Applying this model explains plausibly why Microsoft Windows and Mac OS X can co-exist and sheds light on future implications, RedHat, SuSE and Ubuntu have on the situation. It also explains why commercial computing centres offer LAMP as their primary service platform


Impact of OSS on CSS
Price
CSS
OSS
F2
F1
Quality
Adapted from Jaisingh, et.al, 2006
F1:With ∃ OSS → Q(CSS) drops
F2:With Q(CSS) drops → P(CSS) drops
Illustration 7: Impact of OSS on CSS, adapted from Jaisingh J, et.al., 2006

"*A Strategic Analysis of Competition Between Open Source and Proprietary Software*" takes this idea a step further (Sen R., 2007). Using mathematical models Ravi Sen compares three different market contenders:

- Commercial proprietary software vendors and their respective offering (CSS)
- OSS vendors with support services (OSS-SS) and
- OSS without support service (with OSS only, there is no vendor involved)

The model considers inherent value to users, usability of software, network effects, price, demand and achievable profit. It was designed from CSS's point of view in order to identify risks originating from the existence and growth of OSS.

The **most important finding** is:

> *"The worst-case scenario for both the PS and OSS-SS vendors is when OSS is as usable as PS and OSS-SS. In such a scenario, there is no incentive for the PS and the OSS-SS vendor to enter the software market."*
>
> *(Sen R., 2007)*

This might sound surprising at first. A quick review of the CMS market demonstrates the general validity of this hypothesis. This finding also implies that with a qualitatively satisfactory OSS the **user has no incentive to pay a premium for non-existent advantages**.

Another outcome of this model is that as soon as OSS-SS provides higher inherent value and usability to the customer, **PS will eventually cease to exist**. There is no incentive for users to pay for software that is available for free. This assumes that service fees are basically the same for the two models.

A third conclusion is that OSS being at the same level of usability as PS will eventually force PS out of market. Generally, the price for OSS-SS needs to be less than 50% of PS to be an economically valid alternative. Illustration 8 shows these dependencies and summarises the findings.

A discriminating factor is Sen's model is network effects[43]. It can either extend the range of demand and achievable price for CSS or compete in effect with lower total cost in a business case calculation.

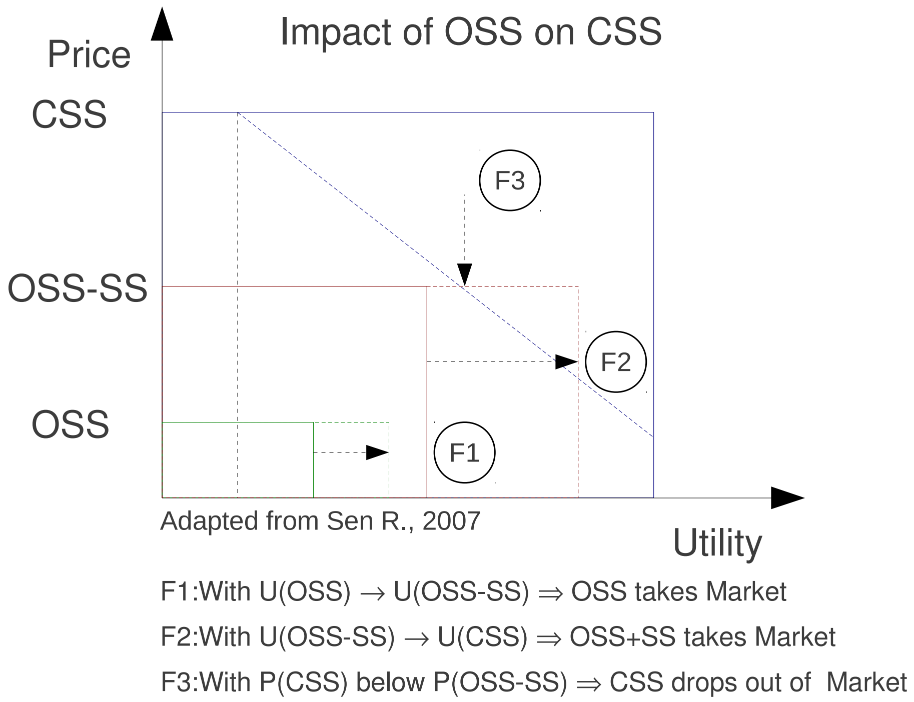

Impact of OSS on CSS
Price
CSS
OSS-SS
OSS
F3
F2
F1
Adapted from Sen R., 2007
Utility
F1:With U(OSS) → U(OSS-SS) ⇒ OSS takes Market
F2:With U(OSS-SS) → U(CSS) ⇒ OSS+SS takes Market
F3:With P(CSS) below P(OSS-SS) ⇒ CSS drops out of Market

Illustration 8: Impact of OSS on Support models and CSS, adapted from Sen R., 2007

From an OSS perspective, network effects can be neutralised in their impact on business value by using and enforcing open standards.

43 What makes Sen's model harder to understand is his interpretation of the term *network effect*. While network effects are generally anticipated to be a positive factor of interoperability, Sen uses the parameter θ in his model to denote the strength of network effects for a given software category. From the examples in Table 3 of his paper it becomes clear that θ is rather a degree of isolation, as the footnote to the table suggests. This is emphasised on page 245 where he confirms that strong adoption of open standards reduces what he understands as network effect, leading to a θ = 0. Thus θ is the degree of freedom or a measure for software lock-in.

Combining Jasingh's and Sen's paper and the fact that development of open source software is an ongoing process to increase quality and utility, the combined theory offers some prospect to OSS. As discussed previously the impact of IP, patents and the development of IP-related laws will influence the pace of this process. It seems improbable though that the process itself will be stopped. This issue and its impact will be dealt in more detail in chapter 3.3.2.

2.3 Critical success factors

To successfully exploit strategic IT management in combination with OSS, critical success factors (CSF's) need to be identified. This chapter collects CSF's contributed from different institutions and research papers. Finally the results of the Viennese migration project will be analysed to extract CSF's and lessons learned.

2.3.1 Fraunhofer Institute

The "*Fraunhofer-Institut für Arbeitswirtschaft und Organisation IAO*" has collected, described and summarised the following advantages and disadvantages of OSS (Renner Th., et.al., 2005):

Advantages	Disadvantages
• Adaptability • Code re-usability • Higher product quality (as opposed to PS) • Vendor independence • Higher product security (as opposed to PS) • Open standards • No license fees	• No claim for warranty • (Sometimes) no support by software developers • Higher cost for training (as opposed to PS) • Uncertain future development • No applications available in certain domains • Sometimes imperfect interoperability with commercial software products

Table 8: Advantages and disadvantages of OSS (translated from original)

The study admits **problems with warranties granted by OSS developers**. This is due to the fact that many globally spread developers had writing access to the source code. Responsibility of specific lines of code cannot be traced down to the originating programmer. Further, private and non-commercial developers are reluctant to accept legal responsibility. However, as the study remarks, proprietary closed source software vendors limit their liability with restrictive end user license agreements.

"By using the software, you accept these terms. If you do not accept them, do not use the software. Instead, return it to the retailer for a refund or credit. ...

A. LIMITED WARRANTY. If you follow the instructions and the software is properly licensed, the software will perform substantially as described in the Microsoft materials that you receive in or with the software.

B. TERM OF WARRANTY; WARRANTY RECIPIENT; LENGTH OF ANY IMPLIED WARRANTIES. The limited warranty covers the software for one year after acquired by the first user. If you receive supplements, updates, or replacement software during that year, they will be covered for the remainder of the warranty or 30 days, whichever is longer. If the first user transfers the software, the remainder of the warranty will apply to the recipient. ..."

(Microsoft Software License Terms, Windows 7 Ultimate, excerpt)

Term A implicitly states that the software will perform as-is[44].
Term B is not in accordance with some domestic laws in the EU[45].

Migrating from PS to OSS and using OSS requires additional training, due to limited distribution of OSS the study continues. The fact that implementing new software requires training is undeniable. Considering the current situation especially in the Office arena, training requirements remain the same as the ones necessary when upgrading Microsoft Office[46].

Imperfect interoperability will always be an issue. **Large PS vendors have neither incentive nor necessity to disclose the inner workings of their software**, data structures or file formats. If they do, they do it on a license based term in order to protect their intellectual property and keep the group of accessors limited. As OSS developers rarely effort to spend license fees, the inner workings can only be identified by reverse engineering techniques. This is prone to errors and changes on the PS vendor side[47].

The availability of advantageous and absence of disadvantageous factors both resemble critical success factors implementing OSS with:

- open standards
- high quality of software

44 Usually, Microsoft products come without printed documentation

45 Warranty periods in Austria are 2 years with a reversed requirement of proof after 6 month

46 The user interface of Office XP, 2003, 2007 and 2010 were all different, requiring substantial retraining

47 As demonstrated with the long lasting problems of interoperability between SAMBA and Windows. Only a court sentence by EU for Microsoft to open up its working specification of file services, the SAMBA team was able to provide a compatible and working software

- secured future development (by a lively community)
- interoperability

seeming the most important ones.

2.3.2 Optaros

A different approach is taken by Optaros (von Rotz B., 2007). Optaros assembled a comprehensive catalogue of OSS products. They related applications according to the following software categories:

- operating systems and infrastructure
- application development and infrastructure
- infrastructure solutions
- business applications

> *"The objective was to select key products/projects in each of the relevant software categories and to describe selected products and benchmark them against what enterprises really need."*
>
> *(von Rotz B., 2007)*

They introduced 4 benchmarking criteria:

- **functionality**: Is functionality comparable to commercial software
- **community**: Is there an active and strong enough community to sustain development
- **maturity**: Is the product stable enough, free of errors and robust
- trend: Is the software still under development? Is it developing in the "right" direction?

These 4 benchmarking criteria as a set were taken to rate the software being enterprise ready with the term "Enterprise readiness" denoting whether an open source product is capable to cope with the needs and requirements of mid sized and large enterprises and organisations. The first 3 benchmark criteria and **enterprise readiness** seems suitable for strategic IT management and they are in line with the list from the Fraunhofer Institute.

Over 260 OSS products were rated that way.

2.3.3 Österreichischer Städtebund

Reifenstein J, et.al, 2004 have assembled the following list of critical success factors:

- detailed requirement specification for client and servers including their ramification
- systematic and stringent project management
- consulting by external experts

- holistic cost estimation (TCO calculation)
- date and time frame of the migration
- professional support during the migration phase (and for the period after the migration)
- documentation
- integration of and participation with users (including training and education)

They include a detailed list of items that need be collected in the requirement specifications within these categories:

- Information about existent and employed software
- Requirements regarding data storage and transfer
- Security requirements

The study emphasises the **importance of interrelation with users of the future system**. Migration from CSS to OSS has been anticipated as being subjected to inferior software in past migration projects. This has also be found a major obstacle as the following statement from a different report indicates:

> *"The biggest hindrance in the whole project and especially in the second phase, is the acceptance of the new software by the staff. The users of Microsoft Windows find it difficult to switch to the new system. They feel that they are migrating to an inferior system and, as a result, small differences are capitalized upon, for example, the fact that the settings for the page layout are in a different location for Open Office makes them feel that the new package is inferior to the well- known Microsoft Office Suite."*
>
> *(Chapter VII, St.Amant K., Still B., 2007)*

One factor the study mentions but underestimates the impact regards the installation and **provisioning of test environments and migration labs**. A test lab should be used to test migration steps before rolling them out on a large scale. It further can be utilized as a training area for users who will be migrated in the near future (acting as an IT simulator for the new environment).

While most of the CSF's in Reifensteins list apply to any genuine implementation project, the issue of user acceptance seems predominant as it touches on psychological aspects of employees.

2.3.4 Dahlander and Magnusson on communities

> *"Using communities is a way for firms to increase the total amount of resources they can draw upon in the innovation processes, but there is at the same time a counter-acting need to appropriate the potential value of an innovation by limiting other firms' access to the same resources and information."*
>
> *(Dahlander L., Magnusson M., 2008)*

It seems critical for the ongoing success of implementing OSS to **tap into** local and international **communities**. We will show later that maintaining close relations to open source communities is essential to build sustainable OSS related business models.

> *"The distributed nature of the innovation puts additional demands on firms aiming to use the knowledge residing in communities for their business purposes, and calls for new means to coordinate and control the development and use of knowledge over time."*
>
> *(Dahlander L., Magnusson M., 2008)*

Maintaining or even participating in a community as an enterprise doesn't come without the risk of the community turning away from supporting certain projects. Dahlander and Magnusson suggest a three step approach to successfully utilize community resources:

1. **Accessing** the community to extend the resources a company has available to solve issues related to OSS. Building up a new community is far more difficult than accessing a pre-existing established community
2. **Aligning** the company strategy with that of the community as companies and developers follow divergent motivation. Strategic alignment includes adopting license practices and influencing the direction of development
3. **Assimilation** by integrating and sharing results. This has been proven successful in the Munich migration with WollMux

These three steps don't go undisputed:

> *"Rather than trying to enforce direct control, firms used subtler means to steer communities in particular directions, and tried to influence community developments by offering incentives to key individuals in the form of payments or fringe benefits for certain tasks, or even salaries to work in leading community roles. Despite the fact that compensation and incentives would seem to go against norms of open source communities, they are indeed used by several firms to influence the direction of development. Firms are aware of the problems involved in using such methods to gain influence,*

but they are outweighed by the perceived benefits.”

(Dahlander L., Magnusson M., 2008)

This emphasises the importance to building up and using a community to successfully implement OSS. It also touches on the critical issue of influence and control by companies investing into OSS development. In chapter 3.3.1 we will discuss the benefits of a lively community maintaining source code up-stream.

2.3.5 Vienna's OSS migration

In the evaluation phase to the migration project of Vienna, the project team found this initial situation (Lutz B. et.al., 2004):

- Overall 1.100 applications were counted, 900 were COTS applications, 200 developed internally by MA14
- 68% of these applications were installed on less than 10 PC's
- 46% or 500 applications were considered to require reimplementation
- 30% or 335 applications were either not analysed or there was no OSS alternative available on the market

In the re-evaluation study about the open source migration in Vienna, the authors report a rather sobering result (MA14, 2009):

- 21.000 PC's are currently installed under the control of MA14 (the organisational unit responsible for IT management)
- 14.000 PC's have OpenOffice installed
- 10% use OpenOffice regularly
- 50% of all Office PC's run software that does not offer open source alternatives (neither a Linux version nor an OSS that covers the same requirements)
- 1.100 software packages are not transferable to OSS, some of which are in-house developed applications

The last point suggests that during the migration phase, **the habit of developing incompatible versions of software continued**.

At the time of writing, the website of the Wienux project (http://www.wien.gv.at/ma14/wienux.html) reports the following technical details about the installed infrastructure. By comparison, the version numbers of the current Ubuntu release are listed as well:

	Wienux	Ubuntu 11.04
Core Operating System	Debian 3.1	Debian 6
Kernel	2.6.11	2.6.38
X-Server	XFree86 4.3	Xorg 1.10.1
Desktop	KDE 3.3.2	KDE 4.6, Gnome 2.32
Office	OOo 1.9.112	LibreOffice 3.3.2
Browser	Firefox 1.0.4	Firefox 4.0.1
Graphics	Gimp 2.2.7	Gimp 2.6.11
Scanning	Sane, Kooka	SimpleScan

Table 9: Comparison of Wienux and Ubuntu version numbers (Source: distrowatch.com)

Distrowatch.com reports the status of Wienux as being discontinued.

There are four lessons to be learned from these findings:

1. The versions of OSS offered by MA14 are outdated. There have been tremendous improvements to Linux as well as OpenOffice or the Firefox web browser both in functionality and usability. It is virtually impossible for an internal IT department to maintain their own open source distribution parallel to operating a complex, distributed and heterogeneous IT infrastructure. The project would have gained more momentum if a **standard distribution** would have been implemented[48]
2. The project team identified a number of applications that were either not available as an OSS version or their portability was uncertain. With over 1.000 applications identified and no prior consolidation of the original environment, any scenario of medium complexity is bound to fail[49]. **Ongoing development of incompatible software** made migration a moving target which **should be avoided**
3. MA14 left adoption of the OSS installation to the benevolence of the user, thus amplifying inertia. Migration projects require a **determination to finalisation**. Lack of it leads to failed projects
4. In the pre-migration analysis, detailed business case evaluation was demanded by Lutz et.al. However, a transparent cost reporting was not established before the migration started. That way, the success of the migration plan or deviations from it could neither be monitored or mitigated. Having established a

48 At the time of first evaluation, SuSE and RedHat were reasonable choices. Currently, Ubuntu is the most prominent and widely accepted distribution according to distrowatch.com

49 Treuchlingen: 50 PC's, thin client based workstations, few business applications
NIVADIS, police of Niedersachsen: 11.500 PC's 23 business applications reduced to a few, all representing successful migrations

transparent cost reporting before migration to OSS, the project team would have identified increased efforts to maintain an up to date combination of tools, shown that duplicate installation effort more resources both in infrastructure and personnel and would have had the chance to achieve satisfactory results.

In this chapter we identified the following critical success factors to OSS implementation projects:

- open standards
- high quality of software
- secured future development (by a lively open source community)
- interoperability
- stakeholder maintenance for increased user acceptance
- up-stream development
- consolidated IT infrastructure
- metrics to measure the above collected CSF's
- the determination to utilise OSS

2.4 Metrics and key performance indicators

So far, we have discussed literature coverage of strategic IT management, distilled strategic benefits from research papers and implementation reports and finally extracted critical success factors.

As the example of Vienna's migration project demonstrates, a set of measurements and operational parameters that can be monitored are premium prerequisites. In this chapter, we try to isolate metrics and key performance indicators (KPI's) to measure successful OSS integration and operation as well as strategic fit.

Tiemeyer suggests a top down approach using IT Balanced Scorecards as the main principle of order. His view of IT-BSC perspectives are (Tiemeyer, E. 2010):

- finance (expense and performance perspective)
- value of IT (integration into business processes, services and operational business perspective)
- architecture and technological standards (product perspective)
- sourcing options (customer and supplier perspective)
- employees (personnel perspective)

Except for the finance perspective, all other perspectives measure both quantitative and qualitative indicators. Architecture can only be measured nominally or ordinally. It can be attributed to investment (making it a financial measure of expenses) or qualitatively described as being modern or obsolete (making it a qualitative indicator). The same argument can be applied to Value of IT, Sourcing and Employees. Influence of OSS communities is not included, due to the general nature of Tiemeyers model.

In order to fit new perspectives and parameters into existing corporate BSC systems, a different approach is chosen, dividing metrics and KPI's into four categories that are more in line with Kaplan / Norton's initial perspectives:

- Financial indicators (measuring attribution of IT to overall corporate success and profitability)
- Quality indicators (measuring secondary effects, including technological quality)
- Environmental indicators (measuring both external community and competition performance)
- Internal indicators

2.4.1 Financial indicators

Two financial indicators that have both operational as well as strategic impact are:

- Total cost of ownership (TCO) and
- Return on Investment (ROI[50])

$$TCO = \sum C_i + \sum_{t=sLC}^{eLC} oE$$

with C_i being capital invested, oE being operational expenditure over the life-cycle of the system analysed (from start of life-cycle sLC to end of life-cycle eLC). Reifenstein J, et.al, 2004 expect TCO attribution to be:

- software 5%
- personnel 65%
- unavailability ... 20%
- hardware 5%
- training 5%

$$ROI = \frac{P}{\sum C_i}$$

With P being profit generated by the system analysed.

These two parameters are in line with common literature (Reifenstein J, et.al, 2004, Renner Th., et.al., 2005, Tiemeyer, E. 2010). Reifenstein et.al. indicate that in TCO calculations, risk emerging from undefined legal situations and open issues on IP shall be included.

50 Some literature label ROI in IT environments ROITI: Return on IT investment to specify that only the IT part is considered. Here we refer to both IT investment and organisational investment if appropriate. In this paper the more general term ROI is used

It should be questioned whether including risk in the TCO calculation is economically and logically correct. An estimation taken from risk management is the annual loss expectancy (ALE[51]):

$$ALE=\sum_{i=1}^{n}(I(O_i)\cdot F_i)$$

with $\{O_1...O_n\}$ being the set of harmful outcome, $I(O_i)$ being the impact (loss) of the individual outcome and F_i being the frequency that such a negative impact occurs (usually given in counts per year).

Potentially negative impacts can outperform TCO by magnitudes. Including the potential risk into a TCO or ROI calculation might render any OSS migration infeasible. Thus it is recommended to calculate risk as a third, separated KPI.

The Fraunhofer institute distinguishes between the following types of cost categories (Renner Th., et.al., 2005):

Introductory cost	Operational cost	Strategic criteria
• Personnel • Consulting • Licences • Training • Migration • Installation • Introduction / initiation	• Personnel • Maintenance • Hardware • Operational training • Update	• Stability • Security • Vendor independence • Usability • Interoperability • Adaptability • Existing know-how

Table 10: Fraunhofer cost categories

It would be advisable to extend the strategic criteria with the following parameters:
- scalability
- openness (opposite to lock-in)
- maturity (as an indicator for quality)

In their final business case evaluation, Renner et.al. chose a net present value based approach based on full and direct cost . Unfortunately, as they used different models for calculation and evaluation their categorisation is not consistent with their previous definitions (see Table 10).

51 There are more accurate algorithms calculating IT risks. Soo Hoo K.J., 2000 has published an algorithm that accommodates for mitigation measures on top of risk calculation. The basic ALE calculation seems suitable enough for our purpose of strategic IT risk evaluation.

Illustration 9 demonstrates the model used in their evaluation:

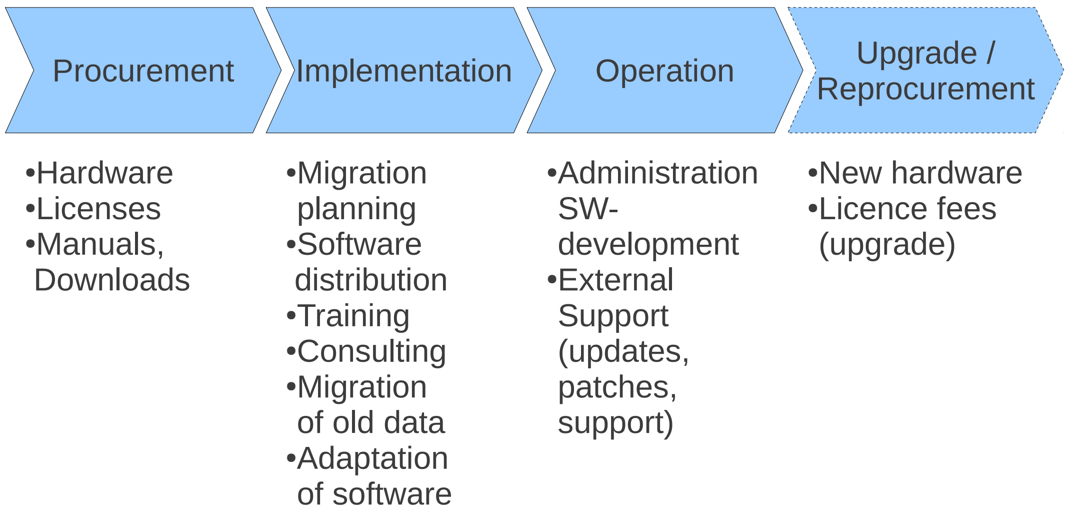


Illustration 9: Software life-cycle and cost drivers (Renner Th, et.al., adapted from Binder S, et.al, 2003)

All cost factors can be calculated as cost per period and differences between periods. In accordance with Renner et.al. calculation of business cases should be direct cost based on net present value. In TCO calculations neither discounting nor accumulation of interest will provide reasonable insight into the actual cost as most parameters (e.g. future interest rates, indirect cost factors and synergetic revenue streams) are subject to speculation more so than planning. TCO calculation should be based on anticipated future expenses.

Maas finds 5 cost categories relevant (Maas W., 2003):

- license cost
- operational expenses
- cost for infrastructure
- support and maintenance cost
- training and educational cost

It is possible to map Renner's categories (as in Table 10) onto Maas's. Renner's categorisation is preferable for project calculation as the more sophisticated and complete model. For monitoring purposes, Maas's model seems sufficient. KPI's can be easily defined, categorised and successively monitored.

In a study about the open commons region in Linz / Austria, Klapf and Plösch provide a catalogue of cost relevant indicators (Chapter 3, Klapf HP., Plösch R., 2010). Their categorisation relating to potentials include the following key indicators:

- cost
- standards

- political aspects and
- availability

As financial indicators, only their cost related parameters are of interest. They are:
- investment and capital expenses
- operational expenses
- cost reduction

Klapf and Plösch's model is a bit oversimplified. Splitting up cost factors in accordance with Maas's model should be a minimum requirement. However, monitoring influence of political aspects and explicitly report on cost reduction seems a valid and interesting approach.

From this categorisation it becomes evident that using OSS from a purely economical and cost based approach does not provide enough benefits and savings as **license and maintenance fees only contributes about 5 – 10% to the total costs**.

2.4.2 Quality indicators

As indicated by Tiemeyers model of IT-Balanced Scorecard perspectives, qualitative measures are vital for strategic IT management.

Quality models related specifically to OSS have only been introduced recently. Early models to measure quality have not been widely adopted. CapGemini's Open Source Maturity Model, OSMM, was not updated since 2003. It used 27 indicators to provide an overall quality indicator. Navica's Open Source Maturity Model measured six different product properties and combined them into a single score. This model was not updated since 2005. Strand L., 2008 cites ISO9000 and IEEE on the definition of quality:

> *“Degree to which a set of inherent characteristics fulfils requirements”*
>
> *(ISO9000 according to Strand L., 2008)*

and

> *“The degree to which a system, component or process meets specific requirements”*
>
> *“The degree to which a system, component or process meets customer or user need, or expectations”*
>
> *(IEEE according to Strand L., 2008)*

As usually there are no predefined requirement specifications for OSS development in a strict sense. The quality is left to the verdict of the observer. Strand defines six basic qualities based on ISO/IEC 9126:

- functionality
- reliability
- usability
- efficiency
- maintainability
- portability

The “Method for Qualification and Selection of Open Source Software” is a well documented set of tools and metrics (http://www.qsos.org/). QSOS offers three levels of evaluation criteria. Evaluation is tool-based, the results can be uploaded to a central database for further querying and interpretation. Illustration 10 displays the general categories used by the model.

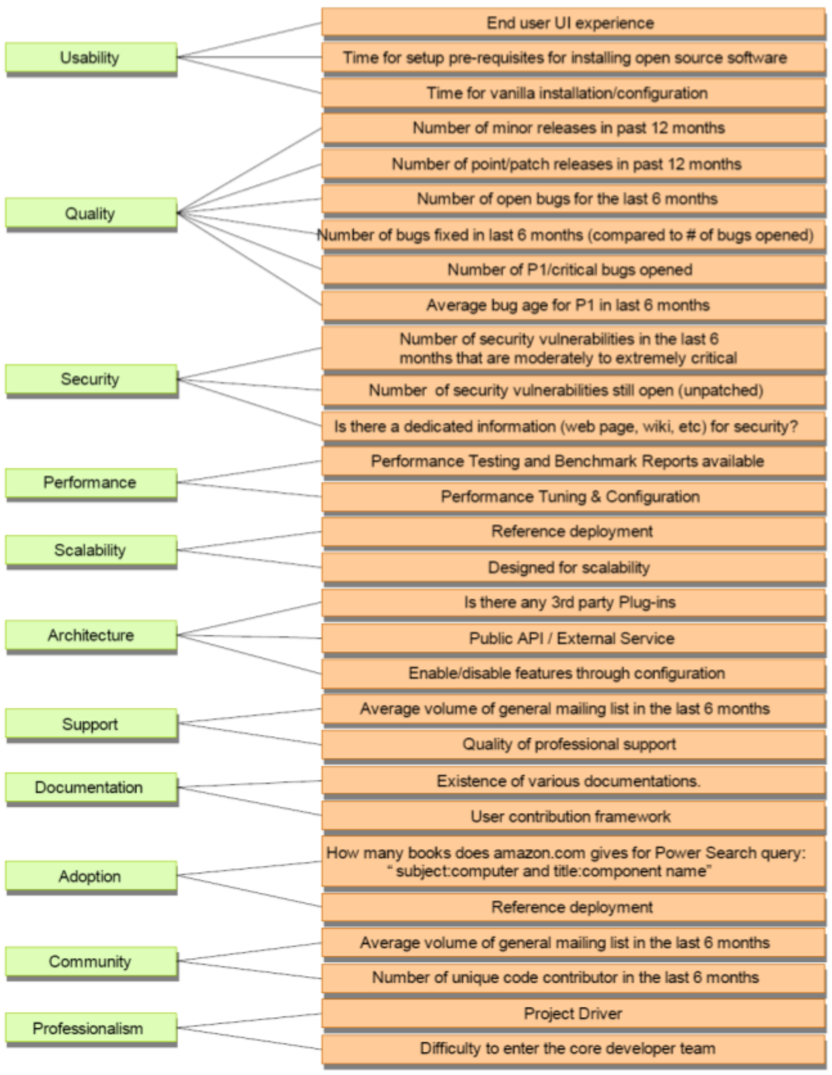

Usability
End user UI experience
Time for setup pre-requisites for installing open source software
Time for vanilla installation/configuration
Quality
Number of minor releases in past 12 months
Number of point/patch releases in past 12 months
Number of open bugs for the last 6 months
Number of bugs fixed in last 6 months (compared to # of bugs opened)
Number of P1/critical bugs opened
Average bug age for P1 in last 6 months
Security
Number of security vulnerabilities in the last 6 months that are moderately to extremely critical
Number of security vulnerabilities still open (unpatched)
Is there a dedicated information (web page, wiki, etc) for security?
Performance
Performance Testing and Benchmark Reports available
Performance Tuning & Configuration
Scalability
Reference deployment
Designed for scalability
Architecture
Is there any 3rd party Plug-ins
Public API / External Service
Enable/disable features through configuration
Support
Average volume of general mailing list in the last 6 months
Quality of professional support
Documentation
Existence of various documentations.
User contribution framework
Adoption
How many books does amazon.com gives for Power Search query: “subject:computer and title:component name”
Reference deployment
Community
Average volume of general mailing list in the last 6 months
Number of unique code contributor in the last 6 months
Professionalism
Project Driver
Difficulty to enter the core developer team

Illustration 10: QSOS metrics (Source: Deprez J.C., Alexandre S. 2008)

“Open Business Readiness Rating”, OpenBRR stems from the same period. It offers 2 levels of categorisation with 29 metrics (http://www.openbrr.org/). The method has not been developed beyond the state of 2006. 1st level categories are:

- functionality
- operational software characteristics
- service and support
- software technology attributes
- documentation
- adoption by community
- development process

This list is in accordance with both the set of strategic activities of chapter 2.1 as well as metrics defined in chapter 2.4.1.

“QUALity in Open Source Software”, QualOSS (http://www.qualoss.org/) has been actively developed until 2009:

> *“The strategic objective of this project is to enhance the competitive position of the European software industry by providing methodologies and tools for improving their productivity and the quality of their software products”*
>
> *(http://www.qualoss.org/about/summary/qualoss-summary)*

The goal of this project is to provide an automated tool that queries and analyses both OSS source code and project repositories in order to rate software quality.

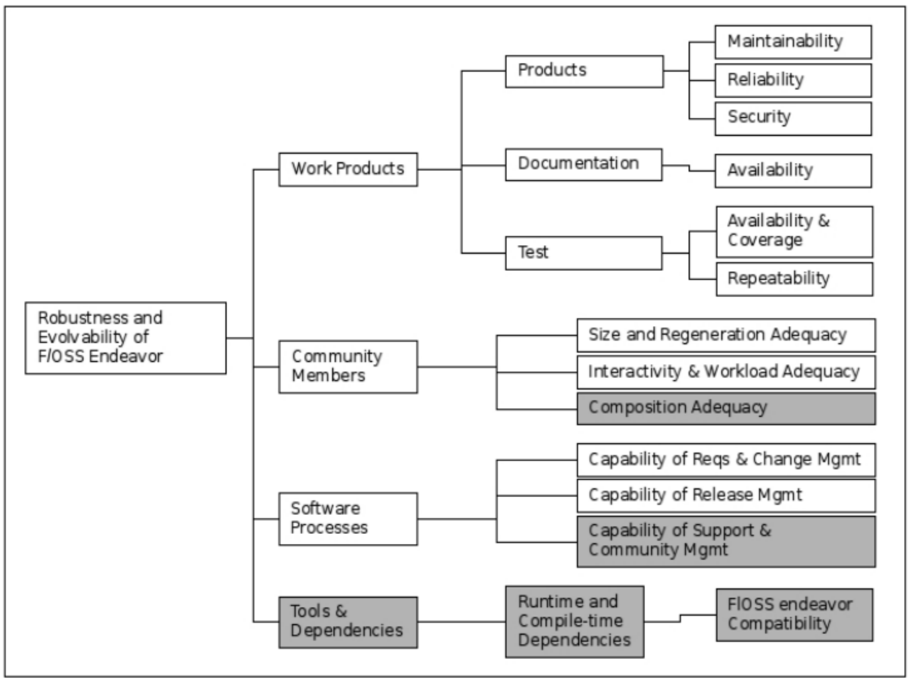

Robustness and Evolvability of F/OSS Endeavor
Work Products
Products
Maintainability
Reliability
Security
Documentation
Availability
Test
Availability & Coverage
Repeatability
Community Members
Size and Regeneration Adequacy
Interactivity & Workload Adequacy
Composition Adequacy
Software Processes
Capability of Reqs & Change Mgmt
Capability of Release Mgmt
Capability of Support & Community Mgmt
Tools & Dependencies
Runtime and Compile-time Dependencies
FlOSS endeavor Compatibility

Illustration 11: QualOSS quality metrics (Source: QuickReferenceGuide_to_StdQualOSSAssessmentMethod_v1_1.pdf)

While quite suitable at the top level, the metrics suggested by QualOSS are somewhat too technology oriented and operational. They might be useful for cross-referencing sets of quality metrics for plausibility and completeness checks.

Finally, Qualipso is a recent standard supported by the EU (http://www.qualipso.org/). Quality is measured using four different abstraction sheets:

- Quality focus
- Variation focus
- Baseline hypothesis
- Impact on baseline hypothesis

Qualipso operates on a catalogue of quality criteria that is separated into two parts:

- product specific (see Illustration 12)
- process specific (see Illustration 13)

For permanent strategic monitoring this model with it's over-complete classification is only of academic value, for reference, consistency and completeness checks. As a means of consistent strategic monitoring it requires reduction to the essential information.

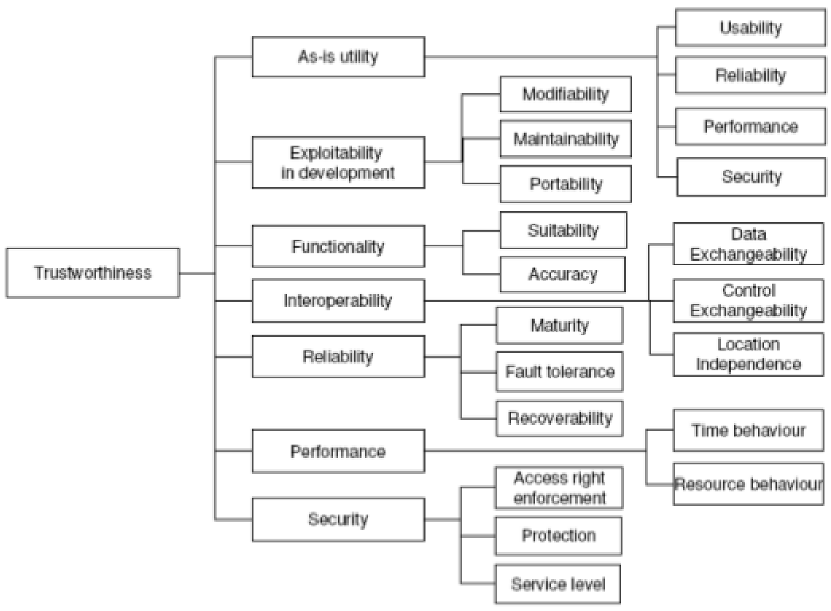


Illustration 12: Qualipso: product specific dimensions (Source: Bianco et.al., 2008)

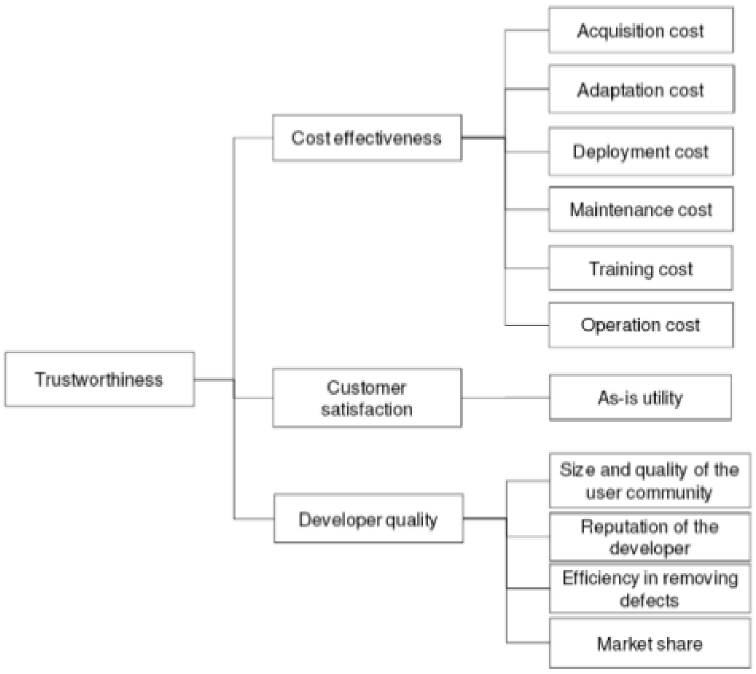


Illustration 13: Qualipso: process specific dimensions (Source: Bianco et.al., 2008)

Klapf and Plösch provide these ideas under the category standards (see Klapf HP., Plösch R., 2010):

- legal aspects
- licenses
- auditable
- service level agreements

They do not specify measurable metrics.

In the last chapter the following quality aspects seemed to be common ground and basis for further evaluation:

- functionality
- reliability
- usability
- efficiency
- maintainability
- adoption and support by a community
- utilisation (usage)

2.4.3 Environmental indicators

Continuing with external environmental indicators, we refer to Klapf and Plösch once more:

- community (is it possible to create a new community, how long does a community exist, how big and active is it and what are their competences)

- political aspects (OSS strategies vs. pressure to implement CSS)
- image (using OSS)

Further environmental parameters relate to customer values, acceptance of OSS and demands for certain OSS functionality.

Another environmental parameter is the business model chosen by software vendors. Koenig describes seven different business strategies (Koenig J., 2004). His work offers a concise yet nearly complete set of different business models. Illustration 14 gives an overview of potential business strategies, backing internal strategies and examples of companies that follow those strategies.

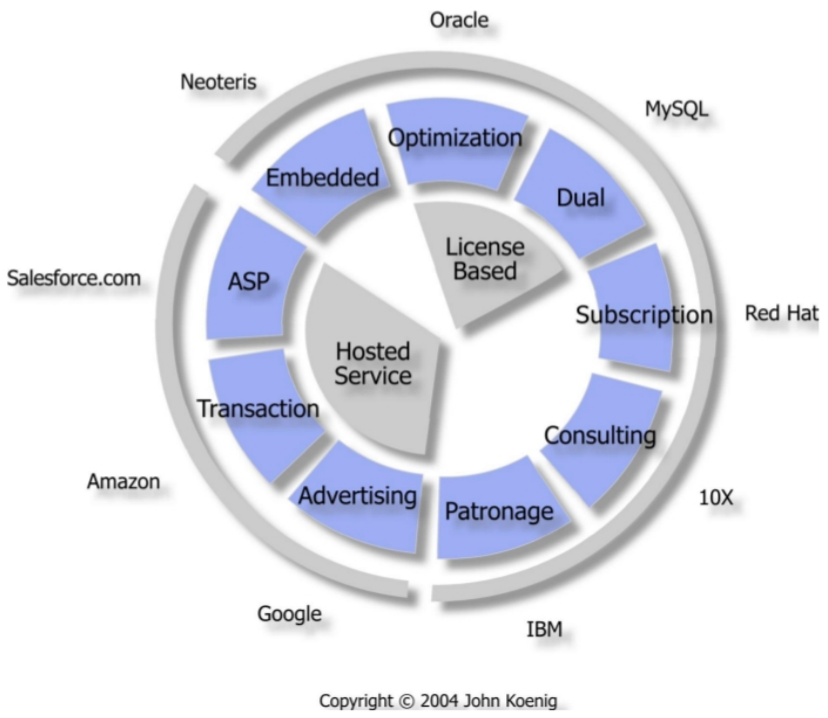


Illustration 14: Business strategies (models) by John Koenig

Most of theses strategies allow only testing on a nominal scale. They are however an important starting point as they reflect the situation of the consumer market.

Koenig's work dates back to 2004. Much has changed since then. MySQL is owned by Oracle, Google has entered the operating system market with Android (a complementary product to embedded mobile devices), Amazon is a provider of cloud based infrastructure services and IBM offers hosted services, consulting and license based systems (CentOS on their P-Series hardware). Two business strategies have appeared on the sceen that require Koenig's model to be updated:

- general services, a combination of consulting, subscription and provisioning
- complementary products, where OSS facilitates as what Erlich and Aviv call widget frosting (Erlich Z., Aviv R. Chapter XV, St.Amant K., Still B., 2007)

Erlich and Aviv have a more simplified view on business models. They distinguish between 4 models:

- **support sellers**
- **loss leaders** (preparing new markets with internally cross-funded OSS)
- **widget frosters** (providing complementary OSS to hardware produced) and
- **accessorising** (selling complementary material to OSS, like books, wearables and other technical gadgets)

Snow Ch.C., et.al., 2009 provide a comprehensive view on innovation adopters:

- **Prospectors**: companies that build their business model on innovation (e.g. Apple Inc.)
- **Defenders**: companies that continually improve on innovations (e.g. Hewlett Packard)
- **Analyzers**: companies that extend the innovation into applied business cases (e.g. Computer Associates)

Snow et.al. analyse the impact, this categorisation has on open innovation, co-operation with communities and the connection to different business models and business strategies. Except for the amendments suggested above, Koenig's model is comprehensive and provides more detailed input into further research. With respect to OSS utilisation, Snow's categorisation does not provide enough discriminating evidence as all three categories may benefit. Koenig's model will be extended and built upon in chapter 3.3.5.

2.4.4 Internal indicators

Some of the above mentioned qualitative indicators relate to internal factors, including developer qualification, motivation and efficiency. There is little material available on these factors partly because the majority of OSS developers act individually only collaborating in their communities. OSS developers that are staffed by large companies like IBM, Oracle or Novell develop software according to established development practices of software engineering and according to corporate business strategies.

There are no metrics mentioned in the material reviewed that covers the inner workings of OSS implementing groups. Conventional methods of software engineering provide some indication of progress. In chapter 3.6 these concepts will be extended into strategic IT metrics.

This chapter covered current research on aspects of open source software with respect to strategic IT management. Strategic IT management was derived from general strategic management. Strategic benefits of OSS were identified and relating research discussed. The **most important strategic benefits** of OSS are:

- Vendor independence
- Availability of source code
- Interoperability
- Efficient use of resources
- Emerging markets and open innovation
- OSS as a long-term alternative to CSS and thus a strategic tool

In order to find metrics suitable to build a set of strategic IT management tools, **critical success factors** were identified and supported by theoretical work. Relevant CSF's identified were:

- open standards
- high quality of software
- secured future development (by a lively open source community)
- interoperability
- stakeholder maintenance for increased user acceptance
- up-stream development
- consolidated IT infrastructure
- metrics to measure the above CSF's
- the will to utilise OSS

Finally a categorisation of **metrics and key performance indicators** has been introduced that allows for integration in an overall IT balanced scorecard. The following KPI's seem suitable to support strategic IT management:

- Financial indicators
  - TCO
  - ROI
  - ALE
- Quality indicators
  - functionality
  - reliability
  - usability
  - efficiency
  - maintainability
  - adoption and support by a community
  - utilisation (usage)
- Environmental indicators
  - community
- Internal indicators
  - none specifically identified

In the following chapter a set of tools applicable to strategic IT management will be introduced that are both feasible to establish and informative to base decisions related to OSS on them.

3 Concepts

Having collected and evaluated literature and material on OSS, this chapter derives methods and tools to answer the research questions formulated in chapter 1.2.

3.1 Motivation

In Illustration 1 the current situation showed that marginal IT contribution to the overall company profit shrinks to zero with increased degree of automation. As companies have reached a high degree of automation already, there is not much incremental benefit gained from introducing new IT and automation systems. The gap between growing prices for licenses and maintenance fees[52] in the market and expected cost cuttings grew to the point where responsible and reasonable operation of IT systems starts to being threatened. New, strategic, intervention to restore IT value contribution is required (Illustration 15).

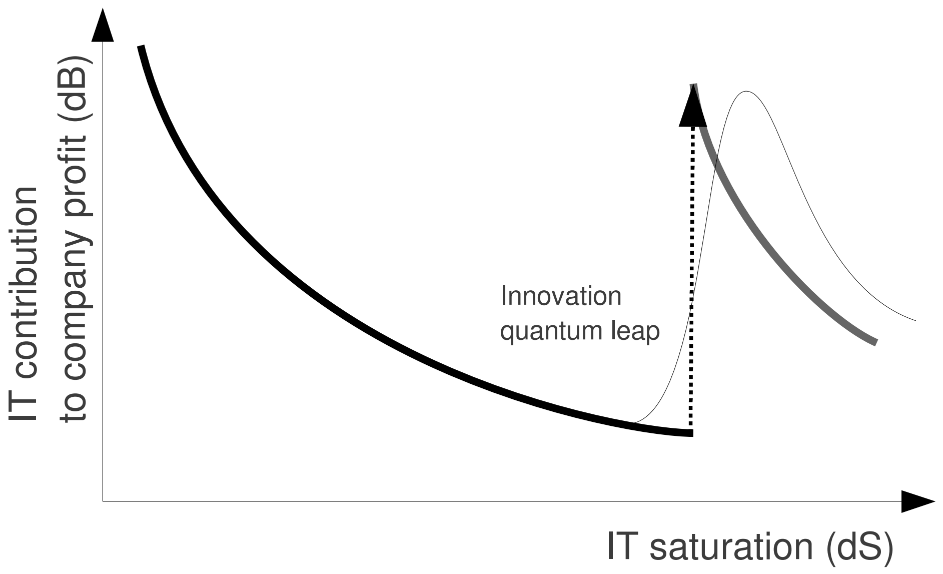


Illustration 15: IT contribution to corporate profit after strategic innovation

One cannot expect that a single innovation will immediately result in a rise in value contribution (dotted line). Gälweiler and Malik explained in their discourse on the substitution time curve how and why introducing new technology or procedures will take some time to take effect (Gälweiler A., Malik F., 2005, p49). This is demonstrated by the thin black line in Illustration 15.

As being observable, IT currently is at the rare end of the graph. New technological approaches are called for to create innovation and thus create new business potential.

52 as described in chapter 2.2.1

Chapter 2 demonstrated that Open Source Software is such an innovation.

3.2 Definitions

3.2.1 Strategic IT Management

Reviewed literature provides a set of circumscriptions of what strategic IT management resembles. This leads to diverse understanding of related issues and consequently, different directions of further activities. This thesis tries to give a concise definition of what constitutes strategic IT management:

> *Strategic IT management is the collection of all necessary processes, tasks and controls to obtain, maintain and sustain existing and new business potentials from an IT perspective and to supervise the effective implementation in order to optimise the value IT generates for the business.*
>
> *As a by-product, strategic IT management controls the efficiency of (operational) IT management.*

Literature emphasises different activities, tasks and focal points. Table 11 gives an aggregate overview of strategic fields of activities. They have been categorised into primary and secondary fields for compatibility with Tiemeyer's (Tiemeyer, E. 2010) and Hanschke's (Hanschke I., 2010) models of strategic IT management.

3.2.2 Up-stream development

In previous chapters, the term "up-stream development" was used to indicate that submission of altered source code into the open source community bears positive long-term effects.

> *Up-stream development refers to the development of software close to the origin of the software. Up-streamed software is attended by the maintainer of the software rather than the instance introducing the software change.*[53]

In this thesis the term is understood as resubmitting and contributing code fragments, patches and applications to the open source community. The impact of up-streaming code will be detailed in the following chapter.

53 This definition is only used for clarifying what this thesis understands under the term "up-streaming" (thus not gray). It is extracted from several definitions available on the web, most prominently Wikipedia.

Strat. fields of activity	Description	Primary	Secondary
IT strategy	Develop IT strategy in accordance with business strategy	✓	
IT architecture	Develop corporate IT architecture (information, application, data, interfaces, software and hardware)	✓	
Sourcing / Procurement	Develop sourcing strategies, procurement regulations and rating of suppliers	✓	
IT controlling	Define IT metrics, measurement procedures, IT BSC, critical success factors, reporting	✓	
IT project portfolio	Manage IT program/project portfolio, requirements, budgets, priorities	✓	
IT marketing	Establish and integrate IT in the company and business departments		✓
IT quality management	Develop quality metrics and rating procedures		✓
Software portfolio	Develop methods, metrics and control mechanisms for software development and selection		✓
IT service level management	Manage IT service maturity and sevice levels		✓
IT security	Manage IT security (legal, technical and commercial)		✓
Legal IT aspects	Manage contracts, copyrights, supplier and customer contracts, licenses and license agreements, maintenance contracts		✓
IT standards	Define and deploy IT standards and IT processes		✓
IT goals	Manage IT goals		✓
IT partner management	Manage supplier and customer relationship		✓

Table 11: Fields of strategic IT management

3.3 Conclusions from Chapter 2

3.3.1 Strategic benefits of up-stream development

Before earning the benefits of submitting source code to the up-stream, it should be clear, that this process requires additional effort to conform to development guidelines of the related open source project[54] as well as to accept basic community rules.

Once the code is accepted, maintaining the code basis still requires some ongoing efforts. Reluctance to do so might result in code elimination[55].

54 Different open source projects adhere to different sets of guidelines and community rules. These apply to programming languages, test- and debugging tools, source code management, bug tracking and code maturity metrics, to name a few.

55 In 2009, Microsoft released approximately 30.000 lines of code into the Linux kernel (http://www.microsoft.com/presspass/features/2009/Jul09/07-20LinuxQA.mspx) to enhance Linux performance on Hyper-V based virtualisation solutions. As this code was not updated and maintained,

This raises the question, where strategic benefits lie if one is to maintain the code submitted.

The first strategic benefit lies in the fact, that **efforts to maintain the code are reduced**. Testing and bug tracking happens by the community. Code fragments are exposed to situations the original developer never intended or even imagined. While the actual code maintenance lies within the instance (developer or company alike) issuing the code, bug fixes, patches or suggestions for improvement come from community members (refer to chapter 2.2.4 on efficient use of resources).

Secondly, source code can and will be **applied to new problem domains** (refer to chapter 2.2.6 on open innovation). As new problem domains will be captured, OSS will be available and used widely thus extending network effects provided by open source solutions[56]. Time to develop new solutions will be shorter resulting in **reduced time-to-market**.

Third, releasing open source code **improves the reputation**, both for companies and individuals alike. This in turn generates future revenue streams.

Companies issuing source code will most likely do so as long as their **core business stays untouched**[57]. Successful methods have been described in chapters 2.2.3 on interoperability and 2.4.3 on business models (Erlich and Aviv). Issuing source code will generate **additional revenue streams** that in most cases outperform the revenue achievable by only vending that software[58].

Individuals benefit from releasing open source as their market value rises, their professional careers improve and their **skills grow**. Releasing open source under permanent scrutiny of an observant community requires to provide top level quality both in methodological approach as well as technological implementation. Successful open source developers have an excellent track record to look upon.

it was eventually eliminated by the Linux kernel maintainers.

56 This can be observed in the area of network security and monitoring tools for example. Network based security is embedded in the Linux kernel. Many network security systems build on top of this infrastructure. Monitoring tools like Nagios offer a plug-in subsystem enabling third party vendors to extend the capabilities of the basic Linux kernel.
Another example is the Eclipse development platform. Originally intended to be a Java IDE with the ability to extend the basic functionality by plugging in new modules, Eclipse now is an application framework that is the foundation for many specific IT solutions.

57 Releasing source code supporting core business functions and knowledge is only feasible if released source code is backed by strong patents. Otherwise this will cannibalise the companies own market and future revenue streams.

58 IBM, one of the worlds largest open source contributor offers Eclipse, a free open source IDE and application framework, additional service, support and value added aplications (e.g. Rational suite) are sold at premium rates.

3.3.2 OSS as a strategic tool to substitute CSS

In chapter 2.2.6 two models were introduced that analysed the impact and strategic consequences OSS has on CSS from a proprietary vendors point of view. Reversing this perspective, a rather strong position of OSS can be derived. This will be detailed in the following argument.

According to Jaisingh et.al (Jaisingh J., et.al., 2006), the final quality of a software product depends on the initial quality of the originating software (presumably code) as well as the resources put into the creation of source code.

$$\frac{dQ_i}{dq_i} \geqslant 0 \quad \text{and} \quad \frac{dQ_i}{dr_i} \geqslant 0 \quad \text{for} \quad i \in c, o$$

for q_i being the initial quality, Q_i the final quality and r_i the resources put into software development. c denotes closed source and o open source.

Initial code quality can be presumed to be fairly equal because open source code is scrutinised by the community, closed source code undergoes professional internal reviews.

Resources put into software directly relate to the payment of software developers. Jaisingh et.al. claim that revenue driven companies invest resources into software development as long as:

- there is a market for the software
- resource input increases the quality of the software (e.g. the software is in a premature state)
- there is sufficient network effect for the software to sustain.

These claims seem reasonable and can be observed in software market development.

If comparable OSS and CSS solutions are in equilibrium, the demand for CSS is:

$$D = D(p, Q_c, Q_o, D^e)$$

with D being the demand function, p the price achievable, Q_c and Q_o the final quality of the competing software and D^e the demand in equilibrium.

$$\text{With} \quad \gamma = \frac{dD}{dD^e} \quad \text{follows} \quad \frac{dp_c}{dD^e} \geqslant 0 \quad \text{and} \quad \frac{dp_c}{dQ_c} \geqslant 0$$

$$\text{but} \quad \frac{dD_c}{dQ_o} < 0 \quad \text{and therefore} \quad \frac{dp_c}{dQ_o} < 0$$

γ denotes the network effect. The price of CSS rises with its quality and demand. However, with rising quality of OSS the demand and thus achievable price for CSS declines.

Finally, if OSS reaches a quality level higher than CSS

$$Q_o \geqslant Q_c \rightarrow Q_c = 0$$

the CSS solution drops out of market and ceases to exist. Customers have no incentive to pay for quality and software that can be acquired for free. It does not even seem necessary for OSS to reach exactly the same level of quality for CSS development to become non-economical.

As defined earlier, the quality of software increases with additional input of resources. OSS resources are voluntary developers, CSS resources are paid software developers.

$$\text{With } \frac{dr_c}{dr_o} \leqslant 0$$

the more developers produce OSS, the less developers are available to develop CSS. While clearly there are forms of co-existence (e.g. companies paying software developers to produce open source code) software developers are a limited resource.

With companies being increasingly reluctant to pay reasonable wages the number of commercial software developers will decline.

$$\lim r_c = 0$$

Jaisingh et.al. conclude:

> *The OSS is thus a passive type of threat to CSS firms. We find that when CSS faces competition only from an OSS, or faces competition from an OSS and another CSS that is not a close substitute, then the incentive of the firms to develop higher-quality products decreases when the quality of the OSS increases. However, when there is competition between two closely substitutable CSS and an OSS, then the incentive of the firms to develop higher-quality CSS increases as the quality of the OSS increases.*
>
> *Jaisingh J., et.al., 2006*

Jaisingh et.al approached the issue from a closed source perspective. Reversing the view provides a bright picture for OSS.

From a historical perspective, **quality of OSS has ever increased**. Development of OSS is rather a process than a project. Developers come and go but communities have far more sustaining life spans. With every new version of any type of software, the existing foundation – core libraries, code samples and fully functional components – will be taken as starting points. This increases the initial quality of OSS beyond what is achievable with closed source developments[59] (Illustration 16).

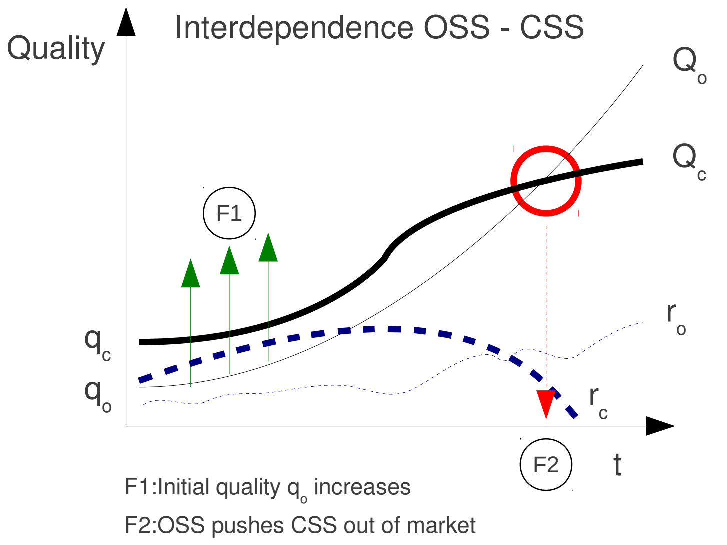

Quality
Interdependence OSS - CSS
Q_o
Q_c
F1
q_c
q_o
r_o
r_c
F2
t
F1:Initial quality q_o increases
F2:OSS pushes CSS out of market

Illustration 16: Impact of OSS quality on CSS quality and life span

As companies decrease wages paid to proprietary software developers there is little incentive for them to stay on the payroll. There are plenty of other business models available to earn a living. Some of them will be listed in chapter 3.3.5, specifically Table 17. This will increase the pool of available open source developers in relation to closed source software developers.

Sen (Sen R., 2007) calculated the utility of software depending on inherent value, market size, added value, price and additional cost due to unusability of inferior software[60]. Illustration 17 demonstrates the influence, increased quality has on the utility of software.

Increased quality of OSS results in a **declining demand and** thus **revenue** for CSS and consequently in reduced input of resources into commercial CSS development. CSS will eventually be repelled into niche markets. Mainstream commodity software

59 Presuming that CSS developers do not copy open source and take a free ride

60 According to Sen's model, CSS is rated as being most usable and having the least additional costs for service and support

based on closed source models will drop out of market due to limited market size, comparable quality of OSS with service and support (or even OSS alone) **limited incentive for vendors to invest into resources to provide competing products** at higher cost. The **utility of OSS will rise** (with service and support) even beyond that of CSS.

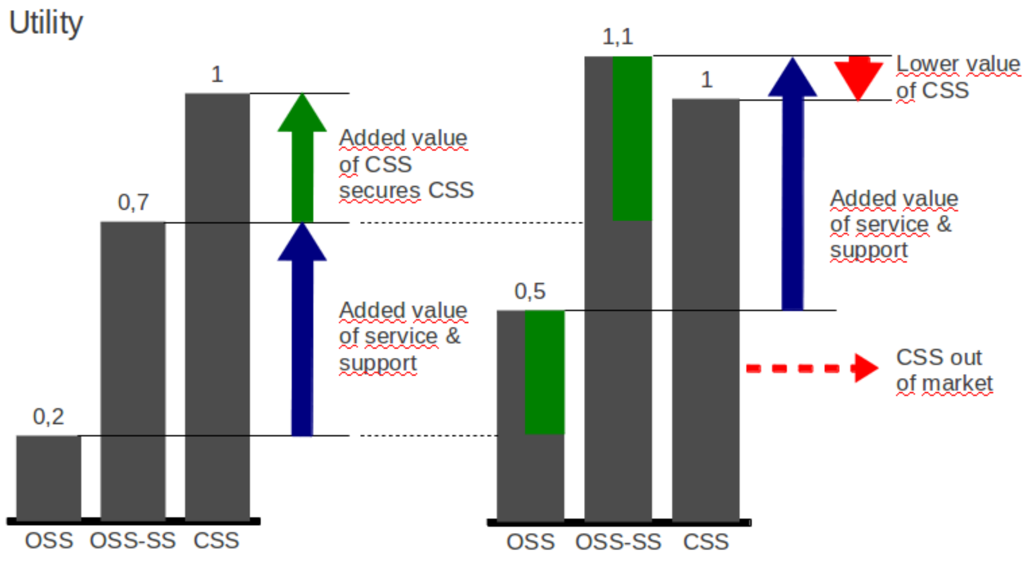


Illustration 17: Development of software utility with respect to increased quality

Sen anticipated the potential threat, OSS poses on CSS. He suggested for CSS companies to gain influence in OSS communities and focus their attention to functionality rather than usability. This way, OSS should never gain the same quality and utility CSS offers. Recent developments have shown however, that OSS developers identified **usability and easy access as a key success factor**[61]. Sen's strategy to infiltrate and distract open source community activities from focusing on the issues important to potential adopters of new software is not showing the desired effect.

It is unlikely that CSS will cease to exist all-together. OSS development follows the mainstream, leaving enough room for highly specialised software categories and niche markets for commercial software development.

However, a **major finding** of this thsis is:

- OSS has the **potential** to become the **pre-dominant model** in the software industry pushing CSS into niche markets

61 Canonical's Ubuntu distribution derived from Debian combines technical excellence of Debian with the simplicity and usability usually associated with the Apple Mac OSX operating system. Ubuntu has gained a leading position in OSS Linux distributions within only 5 years by making Linux easily accessible

3.3.3 Strategic impact of OSS, results from the SWOT analysis

In chapter 2.2 a SWOT analysis based on literature input was carried out. Appendix A provides a summary. In Table 7 we transferred the SWOT analysis into a TOWS matrix to determine basis strategic options. Table 12 extends this view by analysing in detail the implication and derive appropriate (counter-)measures.

TOWS Matrix		O			T				
		Business opportunities	Adoption	Awareness	Legal issues	Community	Technology	Economical	
S	Freedom of use	++			-		?		1
	Evolution of software		++				?		2
	Time, cost and effort	+			-				3
	Quality		++						4
	Advantages to programmers and companies	+	?			?		u	5
	Compatibility			+				-	6
	Availability			+		?		u	7
	Economical	+	+		--				8
	Community			?		o			9
W	Management	-							10
	Quality and security		?				?		11
	Legal issues	?	?		--		-		12
	Compatibility		u				-		13
	Adoption		u				-	u	14
	Economical	u							15
		A	B	C	D	E	F	G	

Table 12: TOWS analysis with impact and action fields

With:

+ (++)	(strong) promotional impact
- (--)	(strong) demoting impact
?	open issue (requires monitoring)
o	neutral outcome
u	unresolved issue (requires effective project management)

3.3.4 Application of strategic benefits to TOWS matrix

In chapter 2.2 the following strategic benefits were identified[62]:

- Vendor independence (VI)

62 Shortcuts were added for quick reference in the following tables

- Availability of source code (SA)
- Interoperability (IO)
- Efficient use of resources (ER)
- Emerging markets and open innovation (MI)
- OSS as a strategic tool to substitute CSS (ST)

Table 12 identified potential application of these strategic benefits in order to utilise OSS to increase IT value contribution to the corporate business value. The following list explains possible applications and their expected impact[63].

SO: **Utilize to best efforts**

Field	Impact	Strat. Benefits
A1	Strong flexible position (business model, technology)	all
B2	exponential growth of users, contributors and fields of application	ST
A3	New product development	VI, SA, IO
B4	Raising quality -> wider adoption -> Jaisingh and Sen will take effect -> CSS will drop out of market	all
A5, B5	3rd world, Asia will offer new applications, Problem: Asiatic languages (including the writing system)	MI, ST
C6, C7	wider adoption, marketing required	MI
A8, B8	lower cost -> wider adoption in use and as sub-product -> increased revenue	ER, MI, ST
C9	Community will demand gratification of development efforts. Idea: Introduction of compensation mechanisms (Google summer of code)	ER, ST

Table 13: TOWS SO impact

63 How to read the table:
Field: Transferred from the TOWS matrix, it denotes the combination of two attributes (e.g. A1 being the combination of Freedom of use and Business opportunities).
Impact: Freedom of use provides a positive impact on new business opportunities as entry barriers and resulting expenses for automation are low. This leads to strong and flexible position in the market, both from a business model as well as technology perspective.
Strategic Benefits: In order to elevate these potentials, the following benefits of OSS can be utilised: Vendor independence allows the company to experiment with new solutions. Achievements can be utilised further, there are little sunk costs. Availability of source code allows the company to extend current solutions to fit ones own needs, a.s.o. Operational implementation depends on the problem domain and company.

ST: **Recover and restore strength**

Field	Impact	Strat. Benefits
D1, D3, D8	Code reuse might trigger law suits, as OSS is open (see Google's Dalvik), constant threat, IP, increases TCO	VI, SA
F1	accept and utilise, technological wealth is positive	VI, SA, IO
F2, E5, E7, E9	Uninteresting[64] areas of software development needs corporate funding, Introduction of compensation mechanisms (Google summer of code)	ER, ST
G6	consolidation before migration	ER

Table 14: TOWS ST impact

64

WO: **Control and monitor competition**

Field	Impact	Strat. Benefits
A10	No business without management, controlling needs improvement	ST
B11	Development of style guides and common UI required -> new business opportunity for GUI designers (see Canonical's UI department)	ER, MI
A12, B12	Legal clarification on OSS adoption required, clear communication, patent pools (for trading), lobbying against SW patents	SA, MI, ST

Table 15: TOWS WO impact

WT: **Rescue and survive**

Field	Impact	Strat. Benefits
F11	Style guides, development standards, see B11	ER, MI
D12	Establish a group of legal and license experts, see A12, B12	SA, MI, ST
F12	Contract management	ST
F13, F14	see F2, E5, E7, E9	ER, ST

Table 16: TOWS WT impact

64 Open source developers tend to focus their work on areas that seem interesting, challenging or simply cool. Other areas, mostly genuine operational problems stay unattended.

3.3.5 Potential users of OSS

Until now, utilisation of OSS was assumed to be equal disregarding the type of business that adopts OSS. This approach ignores varying benefits across disjunct businesses.

An Internet access provider will benefit strongly by implementing OSS to large extents while a law firm might require to use market dominating office software to be compatible with their clients and courts. A hardware vendor might use OSS to provide simple interfacing software and drivers to facilitate adoption of its modules in larger projects. A software developer will choose the platform mostly available in the market and thus vote e.g. for Microsoft and .NET as her foundation.

For further discrimination between different utilisation, the following categorisation of users is offered:

- **Creator:** Software developers create open source software, disregarding whether they use OSS or CSS tools to develop their software.
  e.g. IBM and Apache Foundation are creators of OSS

- **Combiner:** They aggregate pre-existing open source code – sometimes in combination with hardware (as widget frosters) – to provide additional value with new and innovative solutions.
  e.g. Samsung, HTC as well as RedHat or Canonical are Combiners

- **Capitaliser:** (Corporate) users implement OSS (either alone or in combination with support contracts) to support and facilitate their business processes.
  e.g. Munich or the French Gendarmerie capitalise on OSS

These three categories benefit differently from using OSS.

Creators and Combiners can choose from different applications and business models. Table 17 summarises the most prominent business models derived from chapter 2.4.3. Where appropriate, the list was extended and examples added for better understanding.

Abbreviation	Description
Opt	**Optimisation**: Layer based approach to combine underlying modules into a whole (e.g. Oracles hardened Linux on to of Linux)
Dual	**Dual license model**: Providing OSS both under an open source license like GPL and a more restrictive license that allows commercial application (e.g. MySQL, Apples CUPS printing services)
Sub	**Subscription**: Regular maintenance and service on a fixed rate basis (e.g. RedHat, Novell SuSE enterprise services)
Cons	**Consulting**: Providing consulting and services on a one time charge basis (e.g. IBM professional services)
Patr	**Patronage**: Donating personnel and source code to the community, thus directing future development and generating secondary revenue by product sales or enhancing chances of a product to survive (e.g. IBM Eclipse donation, Netscape -> Mozilla donation)
Host	**Hosted Services**: Operating OSS in a controlled environment, protecting IP and offering open API's (e.g. Google Apps, Amazon S3 cloud services, IBM operating centre with CentOS)
Emb	**Embedded Software**: Embedding OSS into hardware, offering additional functionality with OSS (widget frosting) and providing complementary offerings (e.g. TIVO set-top box)
ASP	**Application Service Provider**: Service on top of Hosted Services (e.g. Amazon Cloud Service)
Trans	**Transaction based Services**: Service on top of Hosted Services (e.g. Amazon book sales)
Srv	**Basic Services**: Foundation of subscription and consulting services (e.g. IBM consulting)
Compl	**Complementary Offer**: Part of the internal software development is donated into open source as it does not touch the core business of the donator (e.g. Municipality Munich, WollMux)
Advert	**Advertising**: Providing OSS as a carrier for advertisement (e.g. Google search engine)

Table 17: Business models and strategies (adapted and extended from Koenig J., 2004)

The business models in Table 17 are unlikely to be adopted by Capitalisers. Their choice for utilising depends on a different set of factors.

Most companies use OSS implementations in one form or other. Whether it be a router with a Linux operating system or an Apache web server or hardware embedded firewall, may companies seem to be unaware of their current use of OSS as these systems work without exaggerating conspicuousness.

Table 18 relates strategic benefits identified in chapter 2.2 to company size:

Startegic benefit	Abr.	Small	Medium	Large
Vendor independence	VI		++	+
Availability of source code	SA		?	+
Interoperability	IO	++	++	++
Efficient use of resources	ER	+	+	++
Emerging markets and open innovation	MI	?	+	+
OSS as a business model	ST	x	x	x

Table 18: Applicability of strategic benefits related to company size

65

With:

+ (++)	(strongly) applicable
?	depends on corporate strategies
x	not applicable

A small company will not invest in OSS per se. Lack of resources will force it to choose the set of **applications** according to **best fitting business processes** as well as the availability of external operational IT support. Their decision slack is strongly influenced by software interoperability and to a lesser extent efficient use of resources. It depends on the business purpose whether the company has to co-operate with peer companies in order to enforce open innovation (which might be the case with small companies in a supply chain of highly informal products).

Medium sized companies have a higher incentive to strategically choose OSS. While their ability to utilise source code is limited, their IT needs to be highly interoperable. IT is a factor to differentiate the company from competitors, thus vendor independence is a crucial factor. Lean operation asks for efficient IT systems. OSS with its efficient use of resources (refer to chapter 2.2.4) offers a lot of advantages over proprietary solutions. Co-operations, partnerships and the need for growth is in favour of the strategic benefit of emerging markets and open innovation (MI), even if this is only operational from a business perspective. Investment into IT is under the scrutiny of financial management. **OSS might not be a political option** under these circumstances.
Large sized companies are less dependent on specific vendors. First, they can use their

65 Categorisation of small, medium and large strongly depends on the size of the domestic economy. This paper was written in Austria with a population of 8 plus million inhabitants. Small companies are understood to be between 5 and 20 employees, medium sized companies range up to 500 employees and large companies have 500 and more employees. Very large Austrian companies have 50.000+ employees. The sizing will be varying in different country. Thus the categories are recommendations.

negotiation power to gain profits, small and mid-range companies cannot ripe. They usually have large IT departments flexible enough to adapt to changes in IT system infrastructure. They do depend to a large extend on system interoperability, not only for their vast magnitude of internal systems but also to exchange data automatically with their peers in the supply chain. Efficiency of resource usage is directly related to the TCO of IT thus it is an important factor to consider.

Large companies also might **have the resources** to improve on their IT systems, specifically on their business application software. They usually plan in longer terms and do not have to react immediately to small changes in the market. They can follow corporate strategic plans. Thus they have the **potential to utilise OSS fully** and can effort to **resubmit code into the community**. They have the longevity and financial power to invest into OSS development.

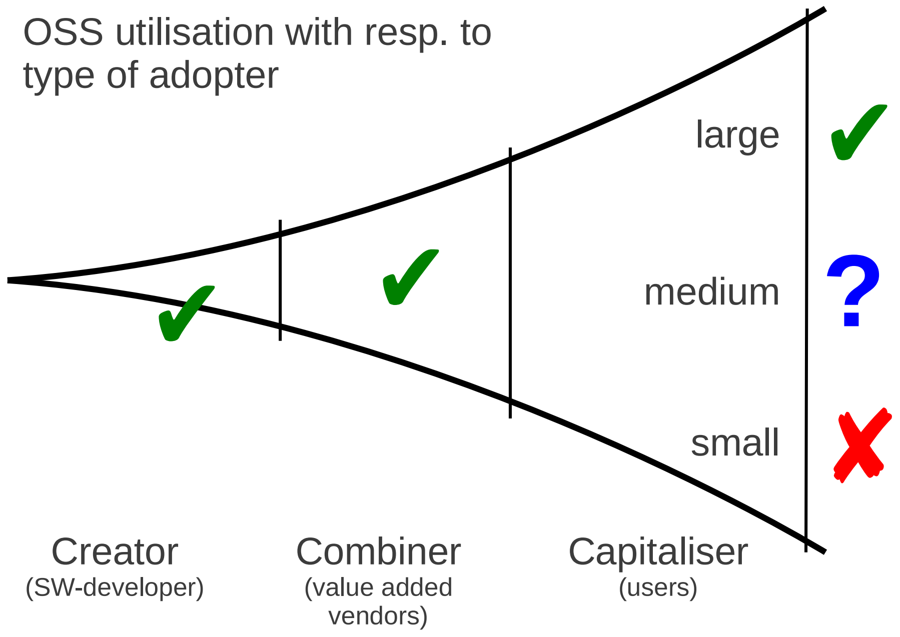


Illustration 18: OSS utilisation with respect to the type of users

Illustration 18 summarises who can utilise OSS.

Common to all areas of OSS adoption is the fact and requirement that **management personnel must be stable and provide continuity**. Many migration projects have failed due to changes in the management team[66].

66 A recent and prominent proponent that re-migrated after a successful migration to OSS is the ministry of foreign affairs in Germany. While Vienna has not announced the end of it's OSS strategy, there is no significant momentum behind it's OSS engagement. Both situations have a change of IT management in common.

3.4 OSS value proposition

In the previous chapter, utilisation of strategic benefits were demonstrated and assigned to strategic fields of activity. Now insight into where and how OSS can contribute to the corporate success is gained.

Open source can contribute in several ways:

- **OSS as a tool with signalling effect**: Open source software is used to automate business and office functions. In the case of strategic IT management, there are some server based tools available under open source license covering many strategic IT management tasks.

  The true value of OSS lies in the signalling effect OSS has on the overall business rather than their superior coverage of application domains. "***If OSS is good enough for strategic IT management, it must be sufficient for everything else***". Tools available support (see Table 19):
  - IT architecture management
  - IT program- and project portfolio management
  - management of software portfolio and software inventory
  - management of IT service levels
  - support for operational IT security and strategic IT security
  - document management for contracts, license agreements, standards
  - IT goal management
  - partner management

- **OS methodology**: Open source development comes along with a set of focused software engineering and project management methods. Life-cycle management with OSS development starts as a rapid prototyping project and goes as far as putting the software onto a staging area (where it is left for the user to download, deploy and operate). Open source project management is a mixture between classic project management with clearly defined deliverables and milestones as well as process management with established procedures and continuous improvement. As the majority of developers work voluntarily, there are no consequences when failing a milestone[67]. To prevent this from happening to often a management of constraints approach is chosen.

  Thus, OS methodology can be seen as **light-weight IT project management** methodology where the problem at hand requires continuous attention,

67 In many OSS development projects, prior to developing the basic functionality, update functions are developed and deployed. That way, any necessary update or modification can be pushed or pulled to the installation

adaptation and process improvement. Strategic IT management is such an activity. It is a regular process, embedded in an overall corporate planning procedure. The level of uncertainty is high (due to very short product life-cycles) and requirements definition vague. Open source methodology facilitates constant small iterative improvements and a permanent flow of information. It can be applied in the following activities of strategic IT management:

- IT strategy (development)
- IT architecture management
- IT program- and project portfolio
- IT marketing
- management of software portfolio and software inventory
- management of IT service levels

- **Open source as a value contribution to IT**: OSS offers some primary and secondary economic benefits[68] that, when accumulated, can provide significant reduction of initial and ongoing expenses, effectively relieving IT budgets and opening up some innovation potentials. Savings are dependent however on several operational implementation factors and environmental parameters. If OSS maintenance is outsourced to a local community, additional business value is created beyond company boundaries.

  OSS can provide software at lower cost, address problem domains that have no economic value to proprietary software vendors (due to low-scale markets) and support rapid prototyping of new solutions for quick testing of advanced business values. Implementation of open source increases network values to business partners. The enforced use of open standards conserves investment into IT infrastructure beyond the usual expected service life. OSS contributes value to the following strategic fields of activity:

  - IT strategy (development)
  - IT architecture management
  - IT quality management
  - management of software portfolio and software inventory
  - IT security management
  - IT standards and
  - strategic IT goals

  as well as IT operations

68 Primary economic benefits are reduced cost for licenses, support and maintenance (in areas where support and maintenance is not operation critical, generic free OSS can be used). Secondary benefits result from lower demand for resources and shorter periods of outages and error recovery

- **Open source business models**: OSS business models can be successfully applied by
  - Creators and
  - Combiners

  Building on Koenig's business strategies, open source business models can act as templates to implement specific processes and procedures to generate additional revenue streams or at least support the creation of business value (refer to Table 17).

  Capitalisers (companies utilising OSS without the need to generate profit from vending it) have to take a different and more diverse approach. Table 18 and the following discourse illustrates this

Strat. fields of activity	Description	Prim.	Sec.	Tool	Method	Value C.	Bus.M	SW product available
IT strategy	Develop IT strategy in accordance with business strategy	✓			✓	✓	Cons.	
IT architecture	Develop corprate IT architecture (information, application, data, interfaces, software and hardware)	✓		✓	✓	✓	Cons. Patr.	interatec
Sourcing / Procurement	Develop sourcing strategies, procurement regulations and rating of suppliers	✓					ASP, Trans.	
IT controlling	Define IT metrics, measurement procedures, IT BSC, critical success factors, reporting	✓					none	
IT project portfolio	Manage IT program/project portfolio, requirements, budgets, priorities	✓		✓	✓		ASP	OfficeWorkbench, GanttPV, GanttProject
IT marketing	Establish and integrate IT in the company and business departments		✓		✓		Advert.	
IT quality management	Develop quality metrics and rating procedures		✓		✓	✓	Cons. Opt.	
Software portfolio	Develop methods, metrics and control mechanisms for software development and selection		✓	✓	✓	✓	ASP	interatec, sonar
IT service level management	Manage IT service maturity and sevice levels		✓	✓	✓		ASP, Cons.	nagios
IT security	Manage IT security (legal, technical and commercial)		✓	✓		✓	Subsc.	Shorewall, Verinice, OpenSIMS
Legal IT aspects	Manage contracts, copyrights, supplier and customer contracts, licenses and license agreements, maintenance contracts		✓	✓			ASP, Cons.	eZ publish, Alfresco, MediaWiki
IT standards	Define and deploy IT standards and IT processes		✓			✓	ASP, Cons.	
IT goals	Manage IT goals		✓	✓		✓	ASP	OpenWeb Bonita
IT partner management	Manage supplier and customer relationship		✓	✓			ASP	compiere, sugarCRM

Table 19: Mapping of strategic IT management to OSS utilisation

Table 19 recapitulates where OSS and OS methodologies can be applied in strategic IT management. Column “Value C.” indicates where OSS can contribute to the value of IT. “Business Model” (abbreviated Bus.M.) refers to Table 17. It summarises the major strategic benefits for Creators and Combiners. Column “Software Available” provides indication where Capitalisers can benefit from specific OSS applications available.

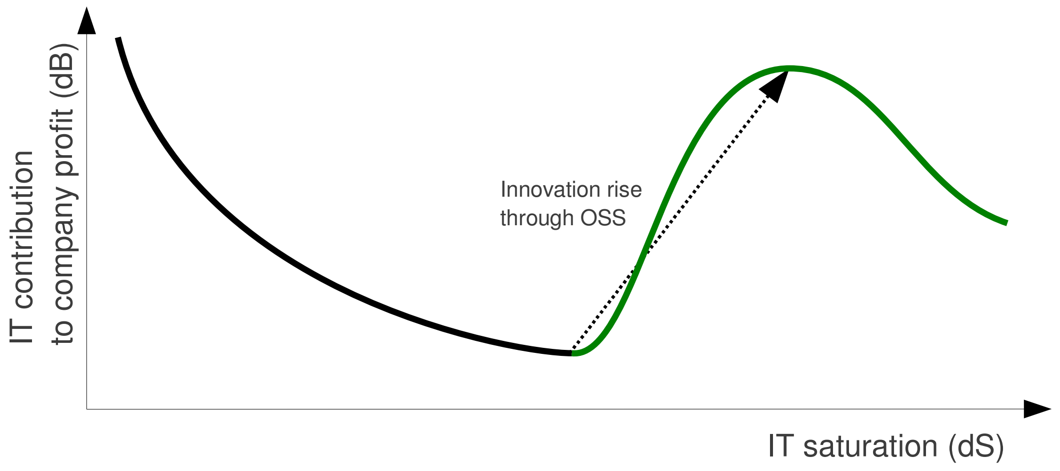


Illustration 19: Possible IT contribution to corporate profit with OSS as strategic option

OSS is no panacea. Introducing or applying OSS on large scales still does not provide the quantum leap as demonstrated in Illustration 15. Voting for OSS as a strategic option will rather show a **slow but steady improvement** as demonstrated in Illustration 19.

With this information at hand, research question Q1 can be answered.

Q1: What is the value of OSS in strategic IT management?

OSS is an accepted alternative in centralised server systems and special purpose embedded appliances. Providers of hosted IT systems offer OSS application stacks – most prominently LAMP – as a convenient service. OSS on the client side currently is limited to specific application domains. The most prominent representatives are Mozilla Firefox and Google Chrome web browsers. OSS is used for widget frosting[69]. A prominent example is routing software on home and small office routers. All these applications are subject of operational IT management.

This thesis showed, through the course of literature research and grounded theory that OSS offers valuable contribution to strategic IT management beyond the operational benefits of isolated, highly specialised IT solutions. OSS contribution to strategic IT management can be categorised into the following categories.

- **Open Source Software as a tool with signalling effect** supports the process of strategic IT management as well as operational requirements. It signals long-term engagement in alternative, business-oriented solutions

OSS tools benefit from advantages described in this paper, like reduced cost, operational efficiency and adaptability to enterprise requirements. While operational in

69 Widget froster: refer to chapter 2.4.3, Erlich and Aviv's business model

themselves, their major benefit lies in their signalling effect.

- **Open Source methodology** is a light-weight project management methodology applicable to a wide range of tasks that are a mixture between processes and projects

Open source methodology involves co-operation and co-development. This gives CIO's a tool at hand to introduce open innovation, address new business opportunities and eventually lift the anticipated value of IT into higher levels of the Forrester pyramid (refer to the introduction for a description of Forrester's pyramid model of IT).

- **Open source as a value contributor to IT** can solve the problems stated in Illustration 1, 2 and 15 respectively. It does not provide a quantum leap in innovation but a slow and steady rise to the business value

Implementing OSS solutions sometimes requires a different approach to traditional CSS system implementation[70]. This can and will enforce the kind of innovation required to increase marginal benefits of IT (Illustration 19).

- **Open source business models** can be adopted – if not fully – then for certain areas and types of the businesses. OSS business models can be successfully applied by
  - Creators and
  - Combiners

- **Up-streaming** internally developed software back into the community will increase the reputation, generate additional business opportunities and relieve the company from tedious parts of software maintenance

An important aspect is the finding in chapter 3.3.2.

- OSS has the **potential** to become the **pre-dominant model** in the software industry pushing CSS into niche markets

If OSS will eventually be the pre-dominant model for software development, this implies that there has to be a migration to this type of software. This migration might be a non-disrupting process from CSS to OSS. As this is highly unlikely, companies might prepare for a system migration sooner or later. Migrating to OSS sooner rather than later will provide these companies a **competitive advantage** in case CSS vendors drop

70 Existence and absence of this kind of unique approach was demonstrated both in the Munich and Vienna migration project. While Munich took a different approach to office automation, Vienna chose a 1:1 migration scenario and failed

out of market. This reduces operational risks and thus lower the amount of private equity reserved to cover this type of risk.

There is enough strategic potential to be tapped into employing OSS. CIO's and IT managers do not have to follow well-worn path's especially under the light of ever increasing economic pressure and shrinking IT budgets.

The next chapter attempts to identify critical success factors to utilising OSS.

3.5 Preconditions for OSS implementation

Implementing OSS from an operational view requires detailed consideration of CSF's as described in chapter 2.3. From a strategic point of view, wide-scale adoption of OSS must be divided into two phases:

- introduction and implementation phase
- adoption and business continuity phase

Strategic adoption of OSS is a long term decision. It will lead IT and the enterprise onto a different road. Adoption of OSS will almost always lead to conflicts between IT- and top management. As there will be problems implementing OSS, IT managers will be facing their competence questioned, allegations about superior proprietary solutions[71] raised and eventually threatened to be recalled.

Professional project planning is required but not sufficient as the many reports on failed migration projects demonstrate. Implementing OSS is a political program. It requires new qualities in communication and IT marketing. The following chapter will discuss preconditions and necessary requirements for successful OSS implementation along the dimensions:

- decisions
- personnel
- software
- hardware

These dimensions go in line with Maas's categorisation in chapter 2.4.1. Quality indicators will be added to these dimensions during the operational phase.

71 CSS vendors employ sales representatives who – in order to meet their individual sales targets – will contact top management to insinuate legal and quality issues if they feel their personal goals being threatened by ongoing OSS implementation projects that substitute the software they represent

3.5.1 Preconditions for the introduction and implementation of OSS

Decisions

- **Strong strategic commitment**: The reason to switch to OSS can be any (combination) of the strategic benefits as described in chapter 3.3.4[72]. As derived in chapter 2.2 any other reason given is either embedded in these strategic benefits or is a tactical reason. In that case, successful adoption of OSS is uncertain[73]

- **Transparent business case**: The business case covers the whole of IT operation. It has to include migration, alternative and opportunity cost, network effects and cost directly induced by this type of program[74]. The parameters of the business case depend on the type of innovation adopter the company is[75]. At least TCO, ROI and ALE should be included in the calculation

- **Roadmap**: A roadmap that illustrates a long range migration plan is critical to demonstrating the activities ahead, progress, milestones and decision gates. The roadmap should be communicated frequently, be freely available and stable regarding concept, content and layout. The roadmap is a **major marketing tool** for the migration team

- **Migration metrics**: A set of KPI's measuring migration progress, success, quality and user satisfaction needs to be put in place. In case of extensive external communication, the number of serviced interfaces and customer/partner satisfaction would seem reasonable to include in the collection

Personnel

- **Continuous stakeholder analysis**: As with any project, a qualitative stakeholder analysis is essential. Additionally this analysis has to be constantly maintained and updated in respect to new stake groups coming in and out and the interaction with them. It is advisable to **maintain the database in a CRM system** and allocate budget for marketing and promotional measures

72 Political influence can be a strong factor, both promotional or deferring. Clarifying the situation in advance is necessary. During the Munich migration, strong intervention by Microsoft forced the project team to re-evaluate the business plan (Stuckenberg B., 2007). The plan was rectified but there was a delay in migration and a phase of uncertainty in the project team nonetheless

73 As soon as the cause for tactical decisions is removed, the cause for migrating to OSS is removed as well. This happened during the Vienna migration

74 Program induced cost includes expenses such as marketing and promotion cost which are required only for this type of venture

75 Innovation adopters are described in Snow Ch.C., et.al., 2009 and referred to in chapter 2.4.3

- **Awareness program**: Special programs to *motivate*[76] users and *communicate benefits* of OSS implementation need to be established and put in place. Without the understanding support of future users, there will be negligence of ongoing efforts and later denial of successful application. This is an ongoing effort that will change only in the methods applied in later phases of migration

- **Peer groups**: A group of early adopters[77] should be established during the initial planning phase of migration. Their responsibilities lie in testing, lobbying for the new solution and establishing links to open source communities. Funding this group should be provisioned for in the business case

- Consulting **Communities**: Establishing links to local communities extends the horizon for new, innovative solutions and potential development and support resources[78]. Strongholds of local communities can be found at universities and educational institutions

Software and Hardware

- **Consolidation of soft- and hardware**: While being an intuitive urge to utilise migration to consolidate application and infrastructure landscape, experience[79] shows that this complicates the migration. Both hardware and software has to be consolidated *prior* to migrating to OSS. This puts problems related to consolidation in the right perspective and does not burden the success of OSS implementation. Consolidating hardware can be achieved by *virtualising* physical hardware where possible. Software supporting *business processes* should be made *web ready*[80].

76 Migration programs will require consequent steps during course. Personal losses will accompany all phases of migration. Without premature motivation and ongoing communication of the benefits, loosers of the migration process will accumulate and form a group of adoption resistors that is hard to control

77 Also called bridge head users, pioneers, power users or test group. Experience shows that allowing this group to form and give it its own name is supportive to the overall process

78 As discussed in chapter 2.2.4, cost for OSS development resources currently is lower than with CSS due to the high percentage of voluntary developers. This resource should be tapped into as long as it is available. Optionally tournaments like Google's Summer of Code to get input to business-related projects could be set up

79 During the Munich migration, problems with document based macros were resolved developing a unified but separated solution. Vienna's migration project failed because the number of non-transferable applications tripled during the course of migration. Having started without these applications in the first place would have increased the chance of successful migration results

80 During the LiMux project, some business applications were migrated to web-based technology in order to prevent migration of client systems (Stuckenberg B., 2007)

- **Inventory**: Create an inventory of hard- and software, including initial and residual value, remaining service period, internal and external maintenance charges. Provide a TCO estimation of the IT infrastructure before migration. This is the bias to which future IT infrastructure will be compared to[81]

- **Test lab**: Establish a test lab that can be used both by administrators and peer groups for experiments and testing of new solutions as well as an educational and training centre for successive user groups adopting OSS

Some lessons learned (but not necessarily preconditions)

- plan for small migration steps
- start with non-critical client applications
- migrate business applications to **web-based technologies**
- **burn bridges crossed** (turn of migrated systems, allow for extended periods of testing but do not allow users to move back, once migrated)
- isolate proprietary solutions (make their use inconvenient[82])
- **do not update software that will be obsoleted** by the migration
- use internal accounting to motivate to use OSS (increase service charges for obsoleted IT services, decrease service charges for new OSS solutions)

3.5.2 Preconditions for the adoption and business continuity phase of OSS

Personnel

- **Information and Training**: Information about the next (operational) steps taken in the migration process is essential to get attention and support from employees. This information should not be provided by the technical migration team but be peer groups that have adopted the change already

Software

- **License sale**: Obsolete licenses that are dispensable after successful migration can be resold in certain countries[83]. This step will not recover past expenses, but fits to the newly adopted strategy of resource efficiency

81 Omission to comply will result in future comparison of cost and benefit of the new solution with a glorified past based on pure speculations

82 Admittedly an extreme suggestion: Install single workstations with specialised software beside the work-group printer

83 http://www.usedsoft.com, http://www.2ndsoft.de, http://www.softstage.de/gebrauchte_software.html

- **Open Standards**: Demand for open standards will increase interoperability and reduce dependency of proprietary interfaces, protocols and consulting services. Provision of automated converters in external communication reduces ongoing incompatibilities with external partners and internal users[84]

- **Up-stream software**: Feeding internally developed code into the open source community and up-stream code relieves the company from tedious maintenance tasks, adds to enterprise reputation, frees internal resources to focus on innovation and open up opportunities for further businesses and co-development

Some lessons learned (but not necessarily preconditions)

- support open communication[85]
- monitor and communicate progress, success and partial failure[86]
- measure performance, quality and cost[87]
- reinvest recovered cost (through license sale) into personnel
- embrace open innovation[88]

We should now be able to answer research question Q2:

Q2: Which basic preconditions must be met to successfully adopt OSS in strategic IT management?

Chapter 3.5 was devoted to this question. Summarising, the following preconditions must be met for a successful adoption:

- Strategic decision and declaration to migrate
- Transparent business case
- Plausible roadmap
- Metrics to measure performance and success of the migration and the target systems
- Stakeholder analysis, monitoring and lobbying
- Established awareness program
- Peer groups acting as early adopters, internal OSS lobbyists and communication bridge-heads to OSS communities

84 Users that have not been migrated yet, have to be considered as having the same status as external communication partners. Treating them as inferior employees will provoke resistance against the migration process

85 This can be done by setting up collaboration web-sites, issue success stories, extend community work

86 Be clear about failure but offer a fall-back solution, work-around or perspective to a solution

87 Refer to chapter 3.6

88 avoid “not invented here” and “been there, done that, got the mug” syndrome

- Established contact with OSS communities
- Consolidated source infrastructure prior to migration
- Economically evaluated inventory of software and IT infrastructure
- Test and training lab(s)
- Permanent and intensive training
- Sale of used licenses and recovery of minor expenditures
- Demand for open standards
- Up-streaming internally developed code

Table 20 summarises the preconditions and their position within the migration phase:

	Phase		
	Initialisation	Implementation	Operation/Maintenance
Decision			
Strategy	Strategic statement, will to migrate		
Business case	TCO, ROI, ALE		
Roadmap	Milestones, decision gates		
Metrics	Set-up	Monitor	Control
Personnel			
Stakeholder	Analysis	Manage	Manage
Awareness	Motivation, Benefits		
Peer groups	Early adopters, OSS lobbyists		
Community	Resource pool		
Information		Success stories	
Training		Test labs	Operational training, manuals
Software			
Consolidation	Legacy apps -> Web		
Inventory	SW inventory, TCO, maintenance cost		
Test Lab		Staging area	Education and user training
Licenses		Install new -> turn of old services	License sale -> fringe benefit, update
	Up-stream code		
Open Standards	Demand	reduced complexity, interoperation	Control, communicate, co-operate
Hardware			
Consolidation	Virtualisation	Dimensioning	
Inventory	HW inventory, resource usage		
Test Lab		Test and educational facilities	

Table 20: Summary of preconditions for successful OSS migration

3.6 Metrics

"You can't manage what you can't measure."

(attributed to Peter Drucker, Robert Kaplan, Edward Deming or TomDeMarco[89])

"Measure what matters. Manage both measurable and unmeasurable elements."

(Dan Galorath, http://www.galorath.com/wp/you-can-manage-what-you-cant-measure.php)

Defining metrics for strategic IT management and related activities is challenging. There is little if any literature around that provides reasonable algorithmic support.

To approach novel situations where limited information is available a three-step approach to create metrics seems advisable:

1. binary selection of existence: yes/no
2. Ordinal classification: T-shirt sizing (with PERT[90]), CMMI[91]
3. interval or ratios: function points, comparison by period

According to chapter 2.4 KPI's are associated to these four perspectives:

- Financial
- External
- Internal and
- Quality

Some key performance indicators seem suitable to be added to a starting set of KPI's. It has been shown by practical experience that less is more. Having a small set of applicable and measurable indicators is preferable to providing a long list of metrics that are hard to collect, require intensive analysis and provide little insight into the operational business.

89 Tom DeMarco has revised this statement in an article for IEEE computer magazine (http://www2.computer.org/cms/Computer.org/ComputingNow/homepage/2009/0709/rW_SO_Viewpoints.pdf)

90 PERT calculation: $Expected = \frac{1 \cdot optimistic + 4 \cdot probable + 1 \cdot pessimistic}{6}$

91 CMMI uses a 5 level maturity scale for software development (CMMI-DEV), service and operation (CMMI-SVC) and acquisition (CMMI-ACQ). Documentation about CMMI can be found at http://www.sei.cmu.edu/cmmi/ a quick introduction and overview at http://en.wikipedia.org/wiki/Capability_Maturity_Model_Integration

In chapter 2.4.1 we suggested three **financial** indicators:

- ROI to measure the profitability and return of the venture
- TCO to measure the overall cost and
- ALE to measure the risk associated with the venture

Environmental indicators may include:

- community interaction indicating how well the company interacts with and benefits from the OSS community
- enterprise readiness indicates the degree, OSS can be adopted in a professional environment

Internal indicators (none where identified in chapter 2.4.4) might be:

- Performance and progress of migration to OSS systems
- A comparison of annual operational expenses. While defined previously that OSS will – if at all – provide only marginal savings, one can still hope for and[92] monitor operational expenditure.

Finally **quality** indicators should at least contain:

- functionality (a measure of how well user requirements are met by OSS)
- maturity (a measure of how stable and reliable OSS applications are) and
- risk (basically the loss expected if OSS does not perform as specified and required, i.e. the annual loss expectancy)

This list certainly does not cover every aspect of strategic IT management and OSS adoption. It does however, provide sufficient overview and insight into overall IT performance. Furthermore these KPI's can be added to a corporate-wide IT-Balanced Scorecard. Table 21 summarises KPI's suggested for strategically monitoring the migration and performance[93] of OSS in enterprises.

92 As the example of Vienna's Linux migration demonstrated, operational cost rose due to dual maintenance requirements

93 KPI P_{mig} can be illustrated as a burn-down chart in order to make the migration progress more transparent

Dimension	KPI	Purpose	Formula	Parameters	Range
Financial	TCO	Overall cost	$TCO=\sum C_i+\sum_{t=sLC}^{eLC} oE$	Ci .. inv. capital, oE .. oper.Expenses, sLC .. start of life cycle, eLC .. end of lifecycle	
	ROI	Profitablity	$ROI=\frac{P}{\sum C_i}$	P .. profit, Ci .. inv. Capital	
	ALE	Equity capital	$ALE=\sum_{i=1}^{n}(I(O_i)\cdot F_i)$	I(Oi) .. Impact, Fi .. Frequency	
External	CA	Community activity	$CA\in[0..4]$	T-Shirt sizing (≤ 4 *), ↑optaros	[0 .. 4]
	Erdy	Enterprise readiness	$Erdy=\frac{\sum Erdy_i}{n}$	Erdy i .. Ent.Read. of applications, n .. num. of apps	[0 .. 4]
Internal	Pmig	Migration Performance	$P_{mig}=\frac{A_m}{t}$	Am .. migrated applications, t .. period	
	AnOpEx	Annual operational expenses	oE/y	oE .. oper.Expenses	
Quality	ALE	Overall risk	$ALE=\sum_{i=1}^{n}(I(O_i)\cdot F_i)$	I(Oi) .. Impact, Fi .. Frequency	
	MAT	Overall maturity	$MAT=\frac{\sum MAT_i}{n}$	MATi .. maturity of application, n .. num. of apps	[0 .. 5]
	FO	Overall functionality	$FO=\sum_{-}\frac{i}{n}$	Func i .. funcionality of applications, n .. num. of apps	[0 .. 4]

Table 21: KPI's for strategic OSS adoption

3.7 Economic benefits

We are now in the position to answer research question Q3.

Q3: What economic benefits can be gained by companies and their environment by using OSS?

This thesis has shown that in order to evade the diminishing contribution of IT to the overall company profit, OSS can offer a large share of economic contributions. With regards to Illustration 19, adoption of OSS is no panacea. It can provide a steady increase in value contribution, necessary for IT to regain its economic value and thus, an enhanced standing of IT within the company.

- The success and value of OSS is not that it is free or cheaper than CSS. The **true value** of OSS lies in the fact that is continuously improved in **quality** and thus provides secondary economic benefits[94] CSS cannot offer

Companies can utilise OSS to release the strategic potentials described in chapter 2.2, most prominently independence from vendors monopolising the software market, elimination of lock-ins, adoption of open standards, utilisation of extended interoperability and consequently reduced cost of license and maintenance fees.

- **Reduced expenses** maintaining legacy systems, **increase IT budgets for** optional and **innovative solutions**

Tactical benefits as mentioned in chapter 2.2.5 can provide additional savings. E.g. enhanced security will reduce the overall risk, the company has to provision[95] for. Support of local partners will strengthen the local economy and provide more flexible resources the company can utilise and thus reduce time-to-market for specific IT solutions and consequently the services the company provides to their customers.

- Positive side-effects of implementing OSS ranging from **reduced risk** to **shorter time-to-market**

Above mentioned benefits are operational side effects of strategical decisions in favour of OSS. They do not come for free – effort has to be taken to harvest economical benefits.

If IT is supposed to **provide competitive advantage, use of standard software** in business critical areas is certainly the **wrong way** to achieve this goal[96]. In this case, OSS is a promising strategic alternative

Finally

- Successful implementation of OSS requires a **stable and determined management team**

The following chapter will critically evaluate the results of the previous chapters and suggest further research.

94 Such as financial resources redirected to local economies, open innovation and co-operation generating successive revenue

95 Risk provisioning is directly related to reserving some of the equity capital and thus reducing liquidity and financial flexibility

96 An often recited misconception is that of business solutions. Business solutions can only be a common denominator but never be adapted to the specific requirements of a unique company

4 Critical evaluation

In the last chapters a set of definitions, concepts and measures were compiled to support the initial claim that OSS provides the potential to increase the value contribution of IT to the overall business. Implicitly the idea of OSS providing an innovation quantum leap (as demonstrated in Illustration 15) was followed.

During the argument of this thesis, the idea of OSS being a panacea for all IT-related problems had to be dropped. OSS offers potential to increase value contribution but this cannot be achieved in a short period of time.

A major statement of this thesis was the claim, that OSS has the potential to outlive CSS software as a pre-dominant development model. The reasoning was laid down in chapter 2.2.6 and 3.3.2 respectively. The hypothesis was based on two mathematical models by Jaisingh et.al. and R.Sen. While Jaisingh's model is mathematically consistent and complete, Sen's model requires some explanation. The combination of both models carried out in this paper requires a complete mathematical proof in order to be fully qualified.

The influence of intellectual property (IP) and software patents and the development of OSS was mentioned but not detailed further. IP and software patents were dealt only from a risk based approach. Risk was provisioned for as a position in the balance sheet[97]. The influence and impact of IP on the long term development of OSS, specifically whether OSS development will happen if any such development is threatened by law suits remains to be considered in detail.

User categorisation and distinction between different sized companies was based on a domestic perspective. This thesis was written with an international market in mind. For some arguments, domestic considerations were taken as a starting point. This requires further analysis.

A set of critical success factors was derived based on previous research. This research showed that many qualitative approaches initiated at the beginning of this century ceased to live after few years of development. It is beyond the scope of this paper to identify whether this was due to the fact that qualitative metrics are covered within ordinary sets of KPI's and thus not specific to OSS or the major initiatives vanished due to lack of financial support. The three qualitative indicators defined in this paper (Table 21) are sufficient as a starting point but require extension.

97 Reserve for impending losses

5 Conclusion

CIO's[98] and IT managers face increasing pressure to reduce cost while providing additional services. This is partly attributed to globalisation and combination for low-cost emerging countries.

> Pay less, get more

Demands for reduced IT costs in combination with increased IT value from both business departments and top management meet with growing expenses to maintain the installed infrastructure come (Illustration 2).

In order to bridge the growing financial gap, CIO's and IT managers must utilise strategic IT management to remedy this situation in long terms and provide strategic potential for growth at reduced cost.

This thesis focused on and answers three research questions (refer to chapter 1.2):

> Q1: What is the value of OSS in strategic IT management?
> Q2: Which basis preconditions must be met to successfully adopt OSS in strategic IT management?
> Q3: What economic benefits can be gained by companies and their environment by using OSS?

This thesis isolated and proved the existence of 6 disjunct strategic benefits OSS offers as opposed to CSS (refer to chapters 2.2 and 3.3ff):

- Vendor independence
- Availability of source code
- Interoperability
- Efficient use of resources
- Emerging markets and open innovation
- OSS as a strategic tool to substitute CSS

98 For abbreviations refer to Abbreviations on page 100

The value OSS provides to strategic IT management can be summarised as (refer to chapters 3.3.1 to 3.4):

- Open Source offers a **light-weight**, distributed, flexible and iterative **project management** methodology
- Adoption of OSS **signals strategic**, long-term **engagement** in open business solutions
- OSS can help IT to get away from providing commodity services to **being a solution partner** and **innovation provider** for the core business
- OSS can be applied, utilised and converted into **business opportunities** all along from the creator of software to the end user

Crucial preconditions were identified, motivated, discussed and consolidated into migration phases (for a complete list refer to chapter 3.5 and Table 20), some of them being unconventional and unique for OSS migration projects:

- **Peer groups** within the company should act as **early adopters**, internal OSS lobbyists and communication bridge-heads to OSS communities
- **Consolidate** source infrastructure **prior to migration**, preferably to a web-based environment[99]
- **Burn bridges crossed**[100]

One might be able to manage what cannot be measured[101] but certainly measurements are required to sell IT services. Chapter 3.6 derives 10 key performance indicators that can be integrated into a corporate-wide Balanced Scorecard system (Table 21).

Chapter 3.7 gave insight into the economic benefits, adoption of OSS provides.

99 Web-based environments are the perfect ecosystems for open source solutions

100 While it is desirable to have a fall-back solution in conventional migration problems, in open source migration projects a more proactive approach will lead (force) to success. The Viennese migration project demonstrated that offering ways back will be used in the face of the slightest headwind

101 Refer to chapter 3.6 “You can't manage what you can't measure”

The **major findings** of this thesis are:

1. Open Source Software will become the pre-**dominant business model** in IT[102]

2. **Success of OSS** will not be attributed to that it's free but provides **higher quality**

3. OSS can be **utilised all along the IT value chain**.
   For End users, **size matters**[103]

4. Successful adoption requires **stable and lasting management** teams[104]

5. **Up-streaming** development is the only way to **secure investment** into OSS[105]

Shall CIO's invest into OSS? From a strategic point of view this thesis wants to encourage and assure CIO's and IT managers to boldly look into OSS as a viable alternative to proprietary software – not only in niche applications but on a large scale basis.

102As the quality of OSS will eventually outperform that of CSS there will be no reason for users to purchase CSS and no incentive for CSS vendors to invest into further development. Niche applications not withstanding. Refer to chapter 3.3.2

103For creators and combiners there are several business models to tap into. For utilising companies capitalising on OSS, the larger the company the more strategic benefits can be taken advantage of. Refer to chapter 3.3.5

104Rapid changes in top management do not provide sustained commitment to migration programs. Refer to chapter 3.3.5

105Up-streaming software back into the community is not a one-time procedure. It requires permanent effort. However, the constant interaction of the community with ones source code provide a magnitude of additional benefits that outweigh the small investment of up-streaming. Refer to chapter 3.3.1

Bibliography in order of appearance

Ferraris P., Porter M. (2009) Global CIO Report, Capgemini Consulting, London W1F0UU

mikey3211 (2007) IT budget under pressure, http://www.techrepublic.com/forum/discussions/

EDUCAUSE center for applied research (2004) ECAR Research Study 7, 2004, 8 IT Cost Management, , USA

Tiemeyer E. et.al. (2010) Strategisches IT-Management, Studienbrief FH-TW MWI 2010, V3.0, Tiemeyer, Wien

Gälweiler A., Malik F. (2005) Strategische Unternehmensführung, 3. Auflage, Campus Verlag, Frankfurt

Maas W. (2003) CIO Querschnittsstudie: Die strategische Option Open Source Software, MCM Instutut St. Gallen,

Enkel E., Gassmann O., Chesbrough H. (2009) Open R&D and open innovation: exploring the phenomenon, Blackwell Publishing Ltd., 9600 Garsington Road, Oxford OX4 2DQ

Chesbrough H.W. (2003) Open Innovation: The new imperative for creating and profiting from techn., Harvard Business School Press, USA

Kell L.T., et.al. (2007) FLR: an OS framework for the eval. and development of management strategies, ICES Journal of Marine Science,

Lettl Chr. (2010) Schöpferische Ergänzung durch Open Innovation, Demokratisierung von LEGO, WU Wien,

St.Amant K., Still B. (2007) Handbook of Research Open Source Software, Technological, Economic,, Information Science Reference, London WC2E 8LU

Gehring R.A., Lutterbeck B. (2004) Open Source Jahrbuch 2004, TU Berlin, Berlin

Hanschke I. (2010) Strategic IT Management: A toolkit for Enterprise Architecture Management, Springer-Verlag Heidelberg,

von Krogh, G., Spaeth S. (2007) The open source phenomenon: Characteristics that promote research, ETH Zürich, Dept. of Management, Technology and, Zürich

Hnizdur S. (2003) The IDA Open Source Migration Guide, netproject Ltd., 124 Middleton Road, Morden, Surrey, SM4 6RW

Lutz B. et.al. (2004) Studie OSS: Open Source Software am Arbeitsplatz im Magistrat Wien, Stadt Wien, MA 14

Reifenstein J., Sallmann R., Haber G., Georgiev E., Zankl W. (2004) Open Source Software - Einsatz in der öffentlichen Verwaltung, Österr. Städtebund, 1082, Rathaus, Wien

Stuckenberg B. (2007) LiMux: Ist Open-Source-Software eine Alternative für die öff. Verwaltung?, Universität Hamburg, Hamburg

MA14, Stadt Wien (2009) Ergebnis der Studie OSS am Arbeitsplatz im Magistrat Wien, MA14, Wien

Gehring R.A., Lutterbeck B., Bärwolff M., (2008) Open Source Handbuch 2008, TU Berlin, Berlin

Heafliger S., G.vonKrogh G., Spaeth S. (2007) Code Reuse in Open Source Software, Management Science, ETH Zürich

Henkel J. (2008) Champions of Revealing - The Role of Open Source Developers in Comm. Firms, Hometown Innovation Automation Inc.,

von Rotz B. (2007) Open Source Catalog 2007 US Version 1.1, Optaros,

Sempert F. (2009) Open Source Software, Aktueller Research, Trends und Perspektiven, Saugatuck,

Crowston K., et.al (2007) Self-organization of teams for free/libre open source software development, Elsevier,

Fosfuri A., Giarratana M.S., Luzzi A. (2007) The Penguin has entered the building, The Commercialization of OSS Products, Universidad Carlos III de Madrid, Calle Madrid 123, 28903 Getafe (Madrid)

Dahlander L., Gann D. (2007) How open is innovation?, Druid Summer Conference 2007,

Jaisingh J., See-To E., Tam K.Y. (2006) The impact of OSS on the strategic choices of firms developing prop. SW, Lancaster University Management School,

Ravi Sen (2007) A Strategic Analysis of Competition Between OS and Proprietary Software, M.E Shape Inc.,

Renner Th., et.al. (2005) Open Source Software: Einsatzpotentiale und Wirtschaftlichkeit, Fraunhofer-IRB Verlag, Stuttgart

Dahlander L., Magnusson M. (2008) How do Firms Make Use of Open Source Communities, ELSEVIER,

Klapf HP., Plösch R. (2010) Studie Open Commons Region Linz, AP5 Kriterienkatalog, ,

Strand L. (2008) FLOSS Quality and Maturity Models, Norwegian Computing Center, Bergen, Norway, Bergen, Norway

Koenig J. (2004) Open Source Business Strategies, John Koenig,

Snow Ch.C., Fjeldstad O.D., Lettl Chr., Miles R.E. (2009) Organizing Continuous Product Development and Commercialization, ,

Illustrations

Illustration 1: IT contribution to company profit..5
Illustration 2: Expected vs. prognosed development of IT expenditures..........................6
Illustration 3: A.Gälweiler: Strategic management and controlling parameters.............13
Illustration 4: E.Tiemeyer: Model for strategic IT management......................................14
Illustration 5: Cost model comparison: PS vs. OSS (Source: von Rotz B. (2007)).........30
Illustration 6: Cost model comparison II (Source: Semper F. (2009))............................30
Illustration 7: Impact of OSS on CSS, adapted from Jaisingh J, et.al., 2006..................40
Illustration 8: Impact of OSS on Support models and CSS, adapted from Sen R., 2007
....41
Illustration 9: Software life-cycle and cost drivers (Renner Th, et.al., adapted from Binder S, et.al, 2003)..52
Illustration 10: QSOS metrics (Source: Deprez J.C., Alexandre S. 2008)......................54
Illustration 11: QualOSS quality metrics (Source: QuickReferenceGuide_to_StdQualOSSAssessmentMethod_v1_1.pdf)........................55
Illustration 12: Qualipso: product specific dimensions (Source: Bianco et.al., 2008).....56
Illustration 13: Qualipso: process specific dimensions (Source: Bianco et.al., 2008).....57
Illustration 14: Business strategies (models) by John Koenig..58
Illustration 15: IT contribution to corporate profit after strategic innovation....................62
Illustration 16: Impact of OSS quality on CSS quality and life span...............................68
Illustration 17: Development of software utility with respect to increased quality...........69
Illustration 18: OSS utilisation with respect to the type of users....................................76
Illustration 19: Possible IT contribution to corporate profit with OSS as strategic option80

Tables

Strategic IT Management: Goals, Activities, Roles, Situations......................................15
5 capital sins of strategic IT management...16
Principles for strategic guidance...18
OSS Strength and Weaknesses...20
Additional strengths and weaknesses (categories added by author)..............................21
OSS opportunities and threats (categories added by author)..22
TOWS matrix derived from SWOT analysis...23
Advantages and disadvantages of OSS (translated from original).................................42
Comparison of Wienux and Ubuntu version numbers (Source: distrowatch.com).........48
Fraunhofer cost categories...51
Fields of strategic IT management...64
TOWS analysis with impact and action fields...70
TOWS SO impact..71
TOWS ST impact...72
TOWS WO impact...72
TOWS WT impact...72
Business models and strategies (adapted and extended from Koenig J., 2004)............74
Applicability of strategic benefits related to company size..75
Mapping of strategic IT management to OSS utilisation...79
Summary of preconditions for successful OSS migration..87
KPI's for strategic OSS adoption..90
Complete collection of SWOT analysis..101

Abbreviations

ALE	Annual Loss Expectancy: metric to calculate the risk associated with a certain activity
API	Application Programming Interface: Convention to access internal functions of software using a defined, stable and tested interface. API's can be viewed as contracts between the provider of the software and later users that wish to extend the software
BSC	Balanced Score Card: Management instrument defined by Kaplan & Norton to monitor and control the company with more than just financial numbers
CIO	Chief information Officer: Executive level management position that is responsible for strategic information and infrastructure management
CMS	Content Management Systems: Systems to produce and publish large websites with volatile content
COGS	Costs of goods sold
COTS	Commercial Of The Shelf software: Synonym for shrink wrapped standard applications
CSF	Critical Success Factor: a parameter that has to meet certain predefined quality standards to facilitate success of operation. No distinction is made between required and sufficient success factors
CSS	Closed Source Software: Software that's source code is concealed, not publicly available and does not conform to the OSI definition
IP	Intellectual Property: The commercial manifest of IP are patents
IT	Information Technology
KPI	Key performance indicator: deterministic value that reflects operational performance of significant systems. Used here in relation to IT systems
LAMP	Linux, Apache, MySQL, PHP (Perl, Python): A computing environment that supports most demands for provisioned computing infrastructure
OSI	Open Source Initiative: a regulative institution defining Open Source Software and maintaining a list of OSS applications
OSS	Open Source Software: Software where source code is available and that conforms to the OSI definition of open source software
PS	Proprietary Software: Software that's source code is concealed, not publicly available and does not conform to the OSI definition. Further the intellectual property of the code belongs to the vendor or has been licensed to be used in products
R&D	Research and Development
RDBMS	Relational DataBase Management System
RFC	Request For Comment: Related set of standards specifically aimed at all areas of information technology
ROI	Return On Investment
SaaS	Software as a Service
SLA	Service Level Agreement: a contract between supplier and consumer of IT services that regulates quality, quantity, price and incentives
TCO	Total Cost of Ownership: The cumulative sum of expenses from the planning, procurement, implementation, adaption, training, operation to the final archivation and termination of operation of a software system
WWW	World Wide Web

Appendix A: SWOT Analysis of Open Source Software

Strength	
Freedom to use	
	Free access to software and source code
	independence from single vendors
	platforms independence
	free upgrade at users own pace
Evolution of software	
	many voluntary developers
	quick bug fixes
	code reuse
Time, cost and effort	
	lower development costs
	quicker bug fixes
	no license fees
	flexible maintenance fees
	code reuse
	no time and budget restrictions during development
Quality of software	
	reduced number of bugs
	user feedback
	constant peer review
	intrinsic code quality (out of programmers self esteem)
	vulnerabilities found quicker
	alternative code distribution channels
Advantages to companies and programmers	
	efficient use of knowledge
	learning by example
	gaining programming experience
	opportunities to collaborate
	rapid development
Compatibility	
	Suitable for certain (not further specified) types of projects
Availability	
	Easy access to OSS
Economical	
	demand driven maintenance fees
	reduced effort to handle license agreements
	beneficial to the local economy
	long term savings
	license model allow flexible development
	low TCO
Community	
	community carries some development cost

Weaknesses	
Management	
	planning and delivering OS community projects is challenging
	coordination and collaboration difficult
	complex resource allocation and budgeting
	fluidity of developers (varying interest)
	existing cost and business models are inadequate
	generating revenue is demanding
	good programmers are hired by PS companies
Quality and security	
	lack of quality documentation
	applications not always intuitive
	no generally accepted style guides
	competition for qualified programmers
	open source invites cyber terrorists
	fewer applications per problem domain available
	Lack of usability
Legal issues	
	legal uncertainties and risk through software patents
	warranties and guarantees
Compatibility	
	limited network effects (no broad installed base)
	limited support for proprietary document standards
Adoption	
	difficult migration path
	slow adoption by companies
Economical	
	OSS model open to free rider syndrome

Opportunities	
Business opportunities	
	Strategic value of OSS indicated
	Adoption by large IT vendors (IBM, HP, Oracle, ...)
	Proven business models available
	OSS acknowledged as business alternative
Adoption	
	Large scale adoption in 3rd world and developing countries
	OSS being complementary to sold products
	Wide adoption by hardware and embedded system vendors
	OSS methods adopted in different industrial areas (open innovation)
Awareness	
	Publicity and awareness

Threats	
Legal issues	
	Changing legal regulations
	Patent laws in EU
	Patent vaults
	Monopolised markets
Community disintegration	
	Outdated, unchallenging projects
	Eliminated personal benefit
	Members outgrow group
Technology	
	Heterogeneous software pool
	License issues
	Specific application domains not covered by OSS
Economical	
	Migration projects infeasible
	Reference migrations unsuccessful

Table 22: Complete collection of SWOT analysis