Value Capture in Open Source Projects

- A case study

Johan Ekholm
Jimmie Johnsson

Department of Computer Science
Faculty of Engineering, Lund University
Ole Römers väg 3
223 63 Lund
Sweden

Department of Informatics
School of Economics and Management, Lund University
Ole Römers väg 6
223 63 Lund
Sweden

Master Thesis Technology, Management – No. 174/2009
ISSN 1651-0100
ISRN LUTVDG/TVTM--09/5174--/SE

KFS i Lund AB
Lund 2009
Printed in Sweden

Abstract

Title: Value Capture in Open Source Projects
- A Case Study

Authors: Johan Ekholm and Jimmie Johnsson

Tutors: Elizabeth Hägg – *Prefect, Department of Informatics, Lund School of Economics and Management*
Martin Höst – *Assistant Professor, Department of Computer Science, Lund Institute of Technology*
Johan Strömhage – *CEO, Purple Scout AB*

Issue of Study: Open Source projects are collaborative processes that generate value for users but generally disallow actors to generate revenue directly from the product that is developed. Nonetheless, many companies are involved in Open Source projects. Therefore it is of interest to study what kind of mechanisms are involved in how companies capture value from Open Source project involvement, and also what kind of negative effects are incurred.

Purpose: The purpose of this study is to answer the following research questions:
How do companies involved in Open Source projects capture value from their involvement?
What kind, and magnitude, of risk does the companies' Open Source involvements entail?

Method: Due to the exploratory and explanatory nature of the research questions, a case study approach was chosen. Four case studies were conducted and the studied objects consisted each of one Open Source project and one company that was participating in that same project. Two to three interviews were performed per case study. Questions and subsequent analysis were based on the structure of the Business Model Ontology. The findings of the study were used to construct a proposition in management literature: the Management Open Source Checklist.

Conclusions: In the studied cases it was found that the companies involved in Open Source projects captured value from complementary offers. These complementary offers included hardware, software and service offerings. The studied companies also gained other benefits that influence revenue streams and cost structure. The extra benefits found in all cases were viral marketing, closer customer relationship and access to external competence. In two

of the cases benefits in forming partnership were also found, and in one case there were benefits in the recruitment processes. In two cases benefits to the company brand were found. In conclusion, the studied companies all capture value primary from complementary offers, but also capture secondary value in form of benefits in other parts of the business model.

Risks with Open Source participation were found in all cases but one. These were: risk of exposing security flaws, risk of being copied by competitors, risk that the Open Source project develops in an unfavorable direction. In all cases the risk were found to be of a lesser magnitude.

The Management Open Source Checklist proposes to be a tool for managers to evaluate Open Source participation from a business perspective. It is based on the findings of this study as well as previous literature but the tool itself has not been validated.

Keywords: Open Source, Business Model, Value, Risk, Community, Standards, Lock-in

Acknowledgements

Researching Business Involvement in Open Source has been an interesting journey for us. The issue spans all the way from ideological programmers, with beards, to hardened businessmen with squinting eyes focused on the bottom-line. We had the opportunity to visit two conferences on the subject, both in Silicon Valley of course, and meet with the people who have these very same questions on top of their agendas. It has been a good incentive knowing that some of the largest technological companies in the world are also racking their brains over this issue.

We would like to thank Purple Scout AB and Johan Strömhage for trusting us with the problem and supporting us all the way. We would like to thank our tutors. Thank you Martin Höst for leading the way through the jungle of scientific methodology. Thank you Elizabeth Hägg for your business-minded advice and sharp proofreading.

We would also like to thank all the interviewees for participating and for giving us their time and knowledge. We would especially like to thank Ron Goldman of Sun Microsystems for interesting discussions about the future of software development.

Lund, May 2008

Johan Ekholm
Jimmie Johnsson

Innehållsförteckning

Glossary

Ajax	A compilation of web technologies used for creating interactive web applications
CMMI	Capability Maturity Model Integration
FSF	Free Software Foundation
GUI	Graphical User Interface
ICT	Information and Communication Technology
IDE	Integrated Development Environment. An application supporting software developers in their work. It normally contains a source code editor, a compiler and a debugger
OSD	Open Source Definition
OSI	Open Source Initiative
RCP	Rich Client Platform. A platform that ease software development
S60	Application Platform for the Symbian Operating System.

Open Source Abbreviations

FLOSS	Free/Libre/Open Source Software
FOSS	Free/Open Source Software
OS	Open Source
OSS	Open Source Software

Licenses

BSD	Berkeley Software Distribution
EPL	Eclipse Public License
GPL	General Public License
LGPL	Lesser Public License

1 Introduction

1.1 History

From early 1960s to early 1980s, computer business hardware was expensive and generated most revenues within the computer business. The first operating systems were developed individually for each computer. (Spiller & Wichmann, 2002) This was done at research stations and for those developers it was natural to keep the code open to make it possible for others to continue building on it. This method of sharing could be compared to the process of academic research. As long as the hardware was that expensive no one saw the value of software. (Orski, 2007)

Over time, there was a need for a uniform system and AT&T developed Unix. Schools were able to use it for a small fee while corporate users had to pay a major fee for licenses. Unix was used for developing Internet technologies and the developers shared code with each other. (Spiller & Wichmann, 2002)

In the early 1980s AT&T changed their licensing policy and only those who paid for the license could use it. This meant that the source became closed and hardware companies such as IBM then started to develop proprietary Unix operating systems. (Spiller & Wichmann, 2002) It was about then software held enough value to be sold separately. By the 1970s it became more common to keep the source code closed and prohibit further distribution by users. (Orski, 2007)

In 1984 Richard Stallman started a project to develop a free alternative to Unix. He named it GNU, which means 'GNU's not Unix' and established a license model to ensure the software being free and open. In 1985 he founded the Free Software Foundation, FSF. (Spiller & Wichmann, 2002)

Free software communities were flourishing, but many vendors did not approve since they found it difficult to create revenues when the source code was freely available. In the early 1990s when the use of the Internet increased, many new Open Source projects started. Linus Thorvalds developed Linux, which is an operating system for desktop computers. (Spiller & Wichmann, 2002)

The Open Source Initiative, OSI, was founded 1997 by Eric Raymond and Bruce Perens. The ambition was to make the term free software more appealing to business as no one thought they could earn money from 'free software'. They agreed to use the term Open Source and the mission for OSI was focused on explaining and protecting the Open Source label. They came up with the Open Source Definition, which is derived from the Debian Free Software Guidelines. (Open Source Initiative, 2009)

During the 1990s many companies started to support Open Source Software projects and Netscape released its Internet browser, Netscape Communicator, as Open Source Software. (Spiller & Wichmann, 2002)

1.2 The Open Source Definition

The term Open Source Software, OSS, was created out of Free Software, FS, to be more appealing to companies. In Europe the term Software Libre, SL, is common. Out of these abbreviations terms have been created like FOSS, Free Open Source Software, F/OSS and FLOSS, Free Libre Open Source Software. (Rosén, 2008) The Open Source Definition by Open Source Initiative is presented in Table 1.

1.	Free redistribution	The license shall not restrict any party from selling or giving away the software.
2.	Source code	The program should include source code, and must allow distribution in source code as well as compiled form.
3.	Derived works	The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.
4.	Integrity of the of the author's source code	The license may restrict source-code from being distributed in modified form only if the license allows the distribution of "patch files" with the source code for the purpose of modifying the program at build time.
5.	No discrimination against persons or groups	The license must not discriminate against any person or group of persons.
6.	No discrimination against fields of end endeavor	The license must not restrict anyone from making use of the program in a specific field of endeavor.
7.	Distribution of license	The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.
8.	License must not be specific to a product	The rights attached to the program must not depend on the program's being part of a particular software distribution.
9.	License must not restrict other software	The license must not place restrictions on other software that is distributed along with the licensed software.
10.	License must be technology-neutral	No provision of the license may be predicated on any individual technology or style of interface.

Table 1 The Open Source Definition by Open Source Initiative (Open Source Initiative, 2009)

1.3 Licensing

Licenses is what makes software officially Open Source. The one that holds the rights to the code, generally the author, can issue licenses. Within the world of Open Source there are several different licenses. This thesis will not explain them all, but for further reading it is useful to understand some of the major differences.

Copyright vs. Copyleft
Copyleft is a generic term for different ways of letting the user modify software and distribute it, as long as the new version has the same license as the original software. If the software is not copylefted the user could modify it and sell it without any problems. (Orski, 2007)

GPL
Richard Stallman created the General Public License, GPL, in 1985 for the GNU-project. The reason was that they wanted to use the same license in their different projects to make sure the source code was open for everyone. If someone else modify or extend software that is licensed as GPL, they have to license their new software as GPL as well. This means that GPL is a copyleft license. (Orski, 2007) It is the most used Open Source Software license. (Spiller & Wichmann, 2002)

LGPL
GPL can make it hard to earn money from the software. Lesser General Public License was created to make it possible to use libraries in building both free and proprietary products. LGPL does not contain rules of which license to use when the software is distributed by someone else. (Orski, 2007) This allows any actor to take LGPL-licensed code, add additional functionality and repackage and distribute it without invoking the license's effects. Changes directly to LGPL-licensed code has to be published however.

BSD
The BSD-license, Bekeley Software Distribution, allows copying and selling of both modified and unmodified software. (Orski, 2007) The documentation materials are also available free of charge. This license makes it possible to trade the product commercially. (Spiller & Wichmann, 2002) There is no requirement that the source code should be open, but the license is often used for open source software. (Orski, 2007)

EPL
Eclipse Public License or EPL, is a weak copyleft license. It is commercial friendly in that it does not require products built on top of EPL code to be licensed in the same way, nor is it required to be disclosed to the public. However changes directly to EPL code is required to be distributed under EPL. An agreement is made with committers to ensure all committed code is put forth under this license. (Eclipse Foundation, 2009)

1.4 Relevance and research goals

As the memory of the dotcom-bubble fades, information technology has seen a steady growth and has become more integral to the way modern businesses is done. The IT-industry has bloomed into a viable business ecology, delivering innovations in communication, data handling and electronic services to an ever-growing market of user and companies. Software development lies at the heart of this technology and some of its specific characteristics seem to be imprinted in the nature of business that evolves around it. Two interesting phenomena are the lock-in effects as well as the Free and Open Source movements. As more businesses and more of our daily lives rely on software, so rises the need to study these phenomena. Several authors have called for increased research in Open Source Software and especially its connection to businesses (Lerner & Tirole, 2002) (West, 2007) (West, 2003).

This thesis proposes to extend the examination of Open Source Software and business. It takes a pragmatic approach in finding the rationale for business involvement in Open Source projects. The first research question is formulated as: *How do companies involved in Open Source projects capture value from their involvement?* Here, the definition of involvement in Open Source projects is that the given company is contributing code to the project on a recurrent basis and that the company has a working relationship with the Open Source community they are contributing to. Value capture is defined as the ultimate economic benefit for the company. It is not this paper's primary intention to examine strategies used, or even the tactics for employing such a strategy. Instead the focus is on the business logic involved in generating benefits, translated to value, for the given company and Open Source project.

The second research question is: *What kind, and magnitude, of risk does the companies' Open Source involvements entail?* Risk is defined as potential threat of negative effects for the company, ultimately resulting in negative economic effects. The rationale for this question lies in the need to compare the benefits with the negative effects. The magnitude of risk is especially hard to estimate since it depends on both the magnitude of negative effects if it occurs, but also the probability of the event to take place. Nonetheless, risk has been cited as the key reason for companies to stay away from Open Source (Rossi, 2004) and therefore it is interesting to examine specifically.

To answer these questions, first the current body of research in Open Source business involvement is reviewed. This is used as a foundation and will be guiding method design, data collection and analysis. It is also the aim of this paper to follow a structured method and analysis, to ensure clarity in reasoning. It is the opinion of the authors that there is some confusion among researchers as what is strategy, or business model, or business processes. Therefore, in this study, the studied companies will be modeled using relevant research development in business models. This model will then be used for analysis.

2 Theoretical Foundation

Open Source has started to get a firm foothold in the research society, so much that it has inspired at least one paper on why Open Source is well suited for research (Krogh & Spaeth, 2007). The authors of that study found that five characteristics of the Open Source phenomenon makes it attractive to research: (1) its impact on economy and society, (2) the fact that it seems to challenge conventional theory, (3) access to source code and development processes, (4) the willingness of the Open Source communities to consider their own working method, and (5) the similarity of the innovation process to scientific research itself.

2.1 Open Source Development

The traditional view on Open Source Development is that it is characterized by a Bazaar-style development mode. In a ground-breaking article, Eric Raymond list principles of Open Source Development. (Raymond, 2001)

A subset of these are presented here:

- Avoid duplication of code by reusing existing software
- Release early and often, and listen to your customer
- Given a large enough co-developer base, almost every problem will be easy to grasp and fix
- Value every contribution to exploit the distributed intelligence of the system

The bazaar metaphor refers to a large, distributed system of developers, all contributing to the same code base. It is characterized by:

a) The absence of centralized decision-making and planning
b) Concurrent debugging and design
c) The integration of users into the development process
d) Self-selection of programmers, in that programmers choose the tasks that best matches their abilities (Rossi, 2004)

The Open Source Development mode can be regarded as a process innovation. It is an experiment of a large-scale parallel development process, with a distributed team of developers. This is enabled by a highly modularized code, meaning that the code is split up in modules with sharply defined functionality and interfaces to one another. This enables programmers to work on one module without disturbing concurrent work on another module. Changes in one module do not disturb the others.(Rossi, 2004)

There has been later research that challenges Raymond's view of the bazaar-type development mode. Studies on the number of contributors to Open Source projects have found that the median is low, ranging from only one (Healy & Schussman, 2003) to four (Krishnamurthy, 2002).

The notion that Open Source development is a collaboration of interacting peers has also been challenged. Empirical studies suggest that the majority of developers are involved in only a few projects (Spiller & Wichmann, 2002), and also that activity is far from evenly distributed but follows power-law-type distributions (Healy & Schussman, 2003). In a case study of two projects, different groups of contributors where found:

1) A core team of developers doing the majority of development
2) Another team of developers, one order of magnitude larger, that fixed bugs.
3) An even larger team of users who reported bugs but did not contribute with any development. (Audris, Fielding, & Herbsleb, 2002)

This changes the view of Open Source Development. Instead of a unorganized mass of contributors that collaborate in a bazaar-like way, the development process seems to be more organized and hierarchical. Contributors are ordered into groups according to competence and skill, and the core design and development is performed by a smaller inner group of people.

2.2 Quality of the Code

Open Source development has been criticized for producing low quality software. This view comes from the opinion that since Open Source development is an un-organized process, relying on the competence of who-ever contributes to the project for what-ever reason, the result can not be as good software as that produced by a dedicated proprietary software development team. At the heart of this view lie concepts such as Capability Maturity Model Integration (CMMI). (Carnegie Mellon Software Engineering Institute, 2009)

CMMI was developed jointly by members of industry, government and the Carnegie Mellon Software Engineering Institute and was sponsored by American military organizations. It is a framework and an approach that aims to improve companies' software development processes. The CMMI framework is based on best practices from traditional industrial manufacturing. One of the main themes is the principle that a company should become more aware of its processes. This is done by modeling the processes and the knowledge in the model is then used to improve the processes in reality. According to the CMMI, doing this will increase the performance of the development organization. (Carnegie Mellon Software Engineering Institute, 2009)

Raymond challenges this traditional view on development processes. In his influential book, two styles of development are juxtaposed: the cathedral and the bazaar style. In the cathedral style there are few developers that work alone in a structured and goal oriented way. This represents the model common used by proprietary software developers. The bazaar style, representing Open Source Development, means that a lot of people develop different parts of the software without any contact with each other

and with different goals. The release should be fast and often, which pleases the contributors since they can see that their efforts are appreciated. (Raymond, 2001)

Raymond argues that if good quality means no bugs, Open Source development is better suited for developing quality code. The code is available at a community and people around the world are inspecting the code and contribute to make it better. This is done without monetary compensation. It would be too expensive for the commercial players to pay their employees to revise the code to the same degree. Therefore, according to Raymond, Open Source Development produces better quality code. He also argues that a community is better at resolving bugs, according to the so called Linus Law: *Given a thousand eyes all bugs are shallow.* (Raymond, 2001)

Even among proprietary developers, voices have been raised against the principles of CMMI. It has been argued that the goal of CMMI to create the very best quality is too expensive and not useful for the customers. Instead of best quality actors can opt for "good enough" quality. In this utilitarian approach some bugs and the associated risk are considered acceptable. The final product may contain several bugs, but as long as the customers experience good quality this is adequate. It is argued that it is the effect of the bugs that matters and the developers should concentrate to fix the bugs that affect the user. (Bach, 1995, 2003)

In a study conducted by Colverity Inc, and funded by the American Department of Homeland Security, the quality of 32 Open Source projects was analyzed. The company used a method that automatically analyzes software code to objectively find code defects. The report concluded that the observed Open Source code on average fared better than baseline code quality. This is an empirical indication that Open Source Development produces good quality software. (Carnegie Mellon Software Engineering Institute, 2009)

2.3 Motivation

Other authors have tackled the theoretical problems posed by Open Source development, related to individual motivation. They ask themselves the question of why people spend time and effort participating in Open Source projects when they are not getting monetary rewards. It has been shown that many of the individual motivational factors can be explained by conventional theory on "career concerns" and organization (Lerner & Tirole, 2002). Motivation for participating is then explained by the need to maximize one's reputation as well as the need for intellectual challenge.

Another way to look at it is to regard the user's need to obtain the code to execute something that is special for that user. It has been argued that heterogenesy among user needs can, in part, explain the rationale for contributing code. (Lakhani & Wolf, 2003) Many other papers explore the plethora of intrinsic reasons for individuals to contribute, such as participating in a gift-giving culture (Raymond, 2001), pure enjoyment (Lakhani & Wolf, 2003) and a sense of identity and belonging (Bergquist & Lungberg, 2001).

2.4 Business Models and Strategy

The theoretical development of the effect of Open Source on business and business involvement seems to develop slowly. This is perhaps due to this area's lack of some of the characteristics that makes other Open Source areas attractive for research. Mainly, business involvement is seldom transparent, and research relies on the good will of businesses that, in many cases, regards lack of transparency as a competitive advantage.

What is called 'business models for Open Source' has been proposed but with little empirical data to back it up (Hecker, 1999) (Krishnamurthy, 2003).

A more empirically robust effort is found in the examination of the Open Source strategy of three major companies (West, 2003). The study focuses on platform strategy and the effect of using Open Source, in parts or in whole. The findings of the study includes the description of the companies' move from proprietary platforms to platforms conforming to open standards and, finally, to open source platforms. As seen in Figure 1, the strategies for open source platforms are found to differ in two dimensions: "opening parts" and "partly open".

Opening parts is explained as the strategy to open parts of the platform architecture while keeping proprietary control of other layers. In this way, some layers or modules of a platform are licensed under a free license and is open for inspection and contributing, while the other layers or modules are closed. As an example, Apple is licensing the core layer of their operating system, Darwin, as Open Source, while keeping the other layers closed. This is aimed to both attain benefits from open source and at the same time retain control over the parts that enable differentiation.

The partly open strategy is linked to releasing the entire architecture under licensing models that retains more control for the licenser. All of Sun's Java technology, for instance, is licensed under Sun's own license, allowing the company to state the conditions. In this way the entire platform is open for inspection and contributing but not as open as the open parts in the former strategy. All the firms researched are of a significant size and all have a history of successfully proprietary platforms. The validity of the theories is thus constricted to that category of company. The question of the benefits of such strategies or any long-term success has not been researched.

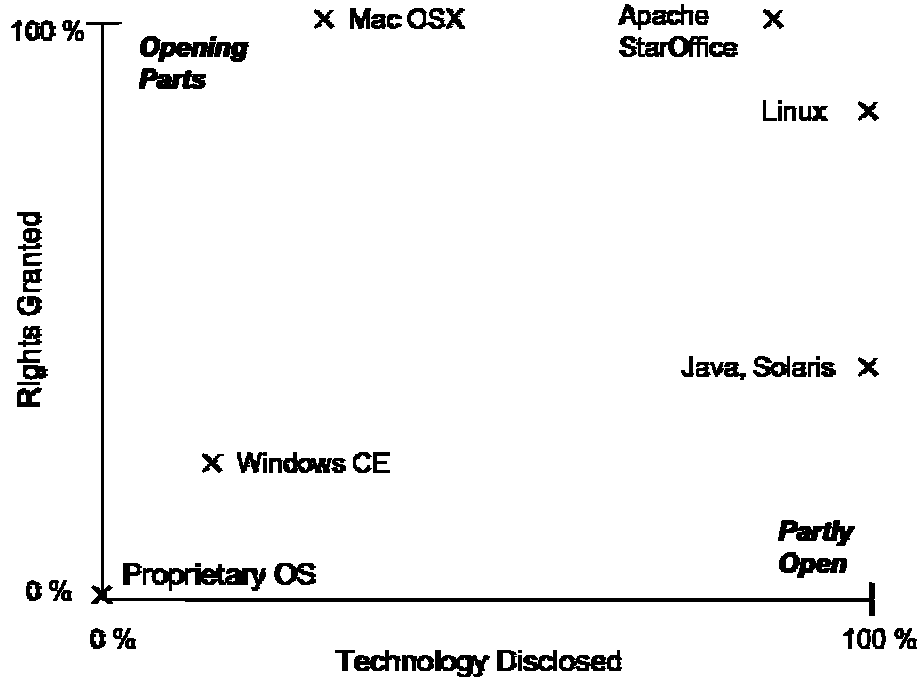

Figure 1 User rights under open and quasi-open source licenses (West, 2003)

A wider take on the Open Source business world is presented in the FLOSS report (Spiller & Wichmann, 2002). It identifies business models used in companies purely based on Open Source, see Figure 2. The definition of a pure Open Source business is that the business would not exist without the open source. The study found business models in two broad categories: Distributors and Service. The Distributor category is further divided into Linux distributors, other product distributors and pure retailers. The Service category contains two subgroups: Service and support providers as well as Open Source development and interest enablers. The former provides service and support to open source software for commercial use. The offerings include technical as well as strategic consultation, integration, support and product specialization. The latter manages marketplaces for development, training material and conferences.

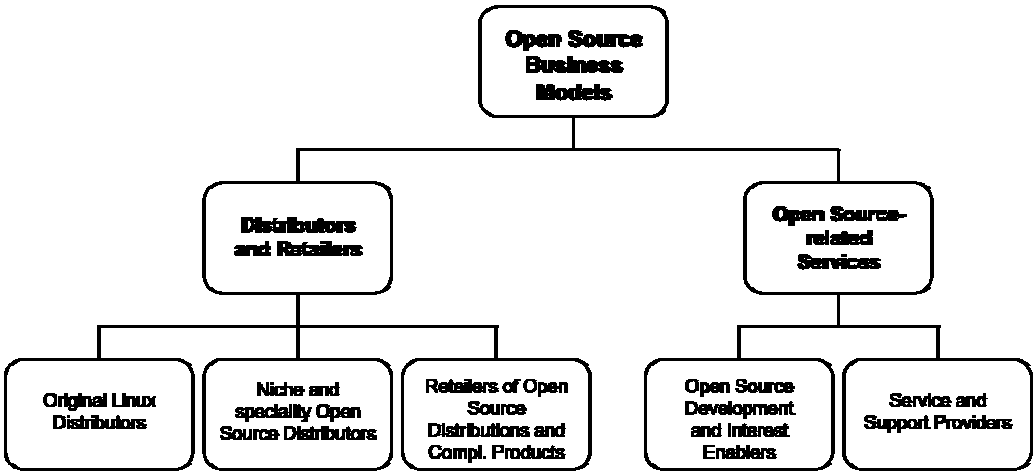

Figure 2 Open Source Business Models (Spiller & Wichmann, 2002)

Another effort includes statistical research on 141 companies, on the nature of their Open Source strategy (Jullien, 2008). This found 4 different groups of strategies regarding Open Source, which can be seen in Table 2: Package, Platform, Architect and User.

The Package strategy involves delivering a whole product based on open source. The benefits of the strategy, compared to a conventional proprietary strategy, is said to be lower cost structure, and greater ease of integration.

The Platform strategy could not be extracted from the data in the study, because of too few relevant responses, but it is included for sake of comparison from the research on platform strategy. The paper argues that the platform strategy may not be available for more than a few companies worldwide, and is thus not readily available for statistical research.

Companies adhering to the Architect strategy are providing individualized IT-infrastructure solutions to their customers. For these companies, there is no substantial revenue gain from using Open Source but rather it is a strategic advantage to be able to access the source code. This gives these companies higher flexibility and control, and therefore possibly more value to their customers.

The last group, Software Users, is comprised of companies that are not themselves involved in Open Source Communities but are simply using Open Source software to provide services. These companies do not gain revenues from the Open Source but rather benefits from a leaner cost structure.

Type of Strategy				
	Package	**Platform**	**Architect**	**Software Users**
Economic Model	Specific offer for software and user assistance	Standard platform and supplementary offer	Provision component-based service	Provision of services based on software (website…)
Competitive Advantage of Open Source	Best relation with clients (user-innovator) and price	Price	Best technical quality, top quality service	Reducing price of the technical solution used, technical independence from editors.
Sources of income in Open Source	Services only. Insurance, assistance, adaption. Temptation to sell software	Sale of supplementary services, customized software aggregation for the platform (possibly outsourced to local distributors)	Same as for standard service companies	None
Open Source Communities	Specific asset for the package community. Heavy involvement (monitoring) of software as cornerstone of the offer. No scattering.	Specific asset for the plate-forme community. Important involvement for key platform components. From zero to weak elsewhere. For local platform distributors, zero involvement.	Potentially strategic complimentary asset. Participation in key component production to be able to contribute.	Complementary asset. No involvement.

Table 2 A synthesis of the link between FLOSS commercial strategy and software development (Jullien, 2008)

The findings from the different studies can be aligned with one another. Jullien's Package strategy coincides with the Retailers and Niche categories of the FOSS Report and the Platform strategy from West and Jullien matches the FOSS report's Linux distribution category (although limited to the Linux platform in that study). The Service and Support Providers category matches the Architect strategy. Thus, even if the empirical studies are few as to this date, they seem to paint a coherent picture.

2.5 Business and Community

Another track is the examination of the relationship between Open Source companies and Open Source communities. A case study on Nordic companies established a typology categorizing the relationship on a scale ranging from symbiotic, through commensalistic, to parasitic (Dahlander & Magnusson, 2005), see Table 3. The different relationships have differing possibilities of controlling the community and

that requires different managerial abilities. The studied companies all offered some kind of product.

Company's relationship to Open Source Community	Description
Symbiotic approach	The firm tries to co-develop itself and the community. In the development of both the firm and the community, the effects on the other party are considered when decisions are taken.
Commensalistic approach	The firm tries to benefit from the co-existence with another entity without harming it. The basic idea in this specific context is to thrive on communal resources that are continually replenished, while keeping the direct involvement in the development of these communal resources to a minimum.
Parasitic approach	The parasitic approach implies that the firm only focuses on its own benefits, without taking into account that its actions might harm the community.

Table 3 Synthesis and typology of approaches. Adapted from (Dahlander & Magnusson, 2005)

2.6 Standards, Network Externalities and Technology Lock-in

During the last two decades researchers in both Economics and Business has recognised the fact that modern high-tech products produce different market dynamics than classical products. *Network Externalities* is one model that tries to explain this. In the model the value of a product that enjoys network externalities is no longer considered independent of other users. The product's value increases with other user's adoption of the product. This also relates to compatibility between products. A product that is compatible, in some sense, with other products can enjoy network effects due to the larger network it creates. Thus there are incentives, from the consumer perspective, to create standards that allow greater compatibility between related products. However this is not always true for the producer. (Farrell & Saloner, 1985) (Katz & Shapiro, 1985)

There are several reasons why these network externalities arise:

1. So called direct effects, or physical effects. When a product is physically connected to other similar products, e.g. a telephone in a telephone network, the value of one product increases with the connection of more products to the network. (Katz & Shapiro, 1985) (Katz & Shapiro, 1986)

2. Indirect, or market-mediated effect. The availability of complementary products increase the value of the product. Examples are software applications available on certain hardware, spare parts or support tools. (Katz & Shapiro, 1985)

3. The availability of know-how and experience, affecting the quality of after-sales services as well as consulting and education. (Katz & Shapiro, 1985)

The presence of network externalities affects the product's or the technology's market. It has been argued that an industry can become trapped in an inferior standard. Under the assumption of incomplete information one model shows that markets are subject to 'excess inertia', and are thus liable to become stuck with a standard that would not be deemed beneficial under circumstances of complete information. The market becomes locked-in to a certain technology. (Farrell & Saloner, 1985)

However newer models have softened this message. When different contracts are taken into consideration, such contrasts as are present in the software industry, the risk for an inefficient outcome is reduced. The different contracts examined where simple market contracts, upgrade contracts and service contracts. Under *simple market contracts* there is no price discrimination between old and new users and new contracts have to become established for each product generation. *Update contracts* on the other hand allow price discrimination between new and old users. Under *service contracts* the price paid once for the product includes the delivery of all future versions. (Thum, 1994)

The influence of actors, notably large corporations, on standards adoption has also been studied. Typically a large corporation has developed or co-developed a new standard and has interest in seeing it becoming widely adopted. It was shown that under influence of network externalities sponsoring is a factor in adoption. When one of two competing technologies is sponsored it is more likely to become adopted, even if it is considered to be inferior. If both of two competing technologies are sponsored the one that is deemed to be superior in the future has a strategic advantage. (Katz & Shapiro, 1986)

Although some markets experience excess inertia, the effect can also be excessive change. A still fully functioning product can be rendered economically useless by the availability of an improved product that is incompatible with the previous generation. There are economic incentives for a monopolistic company to develop new generations

of products which are not backwards-compatible, thus terminating the economic life of previous products and forcing the customers to purchase once more. (Choi, 1994)

Other researchers focus on the managerial perspective. They recognise the fact that certain products can become dominant in a market, lock-in the market, and thus provide greater revenue for the producing company. The dynamics of this is referred to as 'increasing returns' and the causes and strategies for achieving this have been examined. These researchers recognise network externalities as a major cause, but also argue that other factors are influential. (Arthur, 1989) (Schilling, 1999)

These other factors includes:

- *Customer Groove-in*. It requires an investment of time and energy for a customer to learn to use a high-tech product. The customer then experiences high switching cost for other products. (Arthur, Increasing Returns and the New World of Business, 1996)

- *Learning Curve Effects*. Companies that have been developing and manufacturing a product for some time become more proficient with time, thus increasing the distance to competitors. (Schilling, 1999)

- *Up-front Costs*. Many high-tech products have high costs for research and development compared to manufacturing and distribution costs. This means that the company's margins increase with every unit sold. (Arthur, Increasing Returns and the New World of Business, 1996)

- *Signaling Effects*. The size of a product's user base serves as a signal to customers about the quality of the product. The consumer believes that other consumers have already evaluated the performance of different products and settled on the best one. If every consumer follows this logic, no new evaluation will be performed. (Schilling, 1999)

Companies can benefit from these effects by employing different strategies:

- Form alliances with other companies thus promoting network effects.

- Bundling, or linking. The success of an older product can be leveraged to provide a large user base for a new product.

- Promotion and penetration pricing to benefit the initial spreading of the product.

- Scaring off competitors by trying to be perceived as dominant. Competitors will not try to compete in a market where one player is likely to become dominant. (Arthur, Increasing Returns and the New World of Business, 1996) (Schilling, 1999)

2.7 Open Source Value Creation and Value Capture

Value creation is a central concept in both micro- and macroeconomic literature. In management literature, however, a distinction is made between value creation and value capture. The insight here is that although a company creates substantial value for others it is not guaranteed that the value that is created and transacted benefits the company itself. Instead, the value created can be lost to customers, competitors, personnel and/or society. Value capture refers to the act of a company to appropriate value from the value it creates for others. (Coff, 1999) (Lepak, Smith, & Taylor, 2007)

This is a highly relevant difference for businesses that are involved in Open Source projects. The code that is licensed as Open Source is by definition open. Since access to the code means that anyone can compile the code into the final application, the product the code describes is available for anyone without cost. This severely hinders anyone to sell licenses for the product and so the major mechanism for software vendors to capture value is incapacitated.

There is evidence that the Open Source process is well suited for creating value (West, 2003) (West, 2007). The way this works is that Open Source development creates a *positive-feedback adoption loop*, also referred to as "increasing returns" (Arthur, 1996) or "demand side economies of scale" (Katz & Shapiro, 1986). Open Source is especially well-suited for building a large user base and increasing adoption brings positive network effects to the product. The way this influences market mechanics has already been covered under Standards, Network Externalities and Lock-In. A unique effect of Open Source is that increasing adoption also increases the community around the project, thus possibly increasing the value of the product directly by increasing development resources. (West, 2007)

In a recently published article, much akin to this study, the mechanics of value capture in Open Source companies were studied. It was concluded that while value creation is boosted, value capture is a harder task. Businesses were found to capture value from complementary assets, trusting in smaller but more numerous revenue streams. (West, 2007)

2.8 Open Innovation

The Open Innovation Model has been put forth to explain the changing behaviour of some technology companies in the last decade. The model stands in contrast to the Closed Innovation Model that previously explained the success of research companies such as IBM. In the old model research companies gain success from innovations emanating from inside the company. Controlling this intellectual property increased revenue and the revenue was used to reinvest in internal research. This reinvestment resulted in more internal innovations, resulting in more revenues and so large successful companies would become even larger and even more successful. However recent examples cannot be explained by this model, which motivates the need for an Open Innovation Model. Under this model R&D success can be explained by

companies using innovations coming from external resources. The company borders are, by necessity, more porous, allowing the diffusing and absorption of ideas to and from the outside world. The company shares more of its intellectual property and searches the outside world for new ideas that can be used for innovative products and businesses. See Table 4 for a comparison between the Closed and the Open Innovation Model. (Chesbrough, 2003)

Principles of Closed Innovation	**Principles of Open Innovation**
The best people in the field work for us.	Not all the best people work for us so we must find ways to tap into the knowledge and competence outside our company.
We must invent, develop and ship it ourselves to profit.	There is significant value in external R&D; it must be connected to our internal R&D so we can appropiate that value.
If we are the first one to market with an innovation, we will win.	Getting first to market is less important than getting the business model right.
If we create the most ideas internally, we will win.	If we make the best use of internal and external ideas, we will win.
We must control our intellectual property (IP) to prevent competitors to profit from our ideas.	We can profit from other's use of our IP, and we can buy other's IP when it supports our business model.

Table 4 Comparison between the Closed and Open Innovation Model

Lead Users have been proposed as a main factor in how companies work to extract ideas from their environment. This competent user segment often has advanced requirements and much to gain from an innovation in a given field. Access to tools and components, or even to the entire source code as is the case in Open Source projects, allows the lead users to experiment and innovate with a company's product. (von Hippel, 2005)

The following socio-economic factors are used to explain the development toward open innovation systems:

- Labour mobility
- Availability of venture capital
- Number of start-ups
- Density of knowledge workers

The reasoning goes that knowledge is, within many industries, harder to acquire and retain within the company. There are many other competent people outside the company, and acquired competence is likely to move somewhere else within a foreseeable future. These outside people are also able to start their own companies using venture capital and monetise their own innovations. The Open Innovation Model tries to explain how companies work in this environment to profit by the innovation process emerging at their borders. (Chesbrough, 2003)

2.9 Business Model Ontology

As can be seen in the papers grappling with business models for Open Source, there is general confusion as to what a business model actually is. When explaining how a given business works, quite often only some strategic advantages are mentioned (Hecker, 1999). Obviously if one wants to examine how one business generates value it will be useful to model that business. A business is a complex system of human and financial interactions and thus the model should provide a simplified and formalized view of the complex reality. A comprehensive view on business modeling is provided by the Business Model Ontology, BMO, (Osterwalder, 2004). It defines a business model as *the translation of a company's strategy into a blueprint of the company's logic of earning money* (Osterwalder, 2004)

The Business Model Ontology is presented in Table 5-6 and Figure 3 and summarizes all relevant objects that a business model can contain as well as the relations between them. It is designed to capture the essence of modern ICT businesses and is formalized to such a degree as to enable understanding in a computer language (although this is an optional feature). The Business Model Ontology defines four main areas, or pillars, of the model: Product, Customer Interface, Infrastructure Management and Financial Aspects. The three first main areas correspond to the questions: *what?*, *who?* and *how?* respectively. The last main area, the Financial Aspects, underlies all the other areas and represents the financial effects of the others. The four main areas contain nine elements that, taken together, completely model a business.

Business Model Area	***Elements in Business Model Ontology***	***Description***
Product	*Value Proposition*	The Value Proposition is an overall view of a company's bundle of products and services that provide value to its customers.
Customer Interface	*Customer*	The Customer are the segment of customers a company wants to offer value to.
	Channel	A Channel is a means of contacting and interacting with the customer.
	Relationship	The Relationship describes how a company interacts with and relates to its customers.

Table 5 The nine elements of the Business Model Ontology. Part 1 (Osterwalder, 2004)

Business Model Area	*Elements in Business Model Ontology*	*Description*
Infrastructure Management	*Value Configuration*	The Value Configuration describes all the activities that are needed to create the company's Value Propositions.
	Capability	A company's capability is the result of its combined resources that are needed to execute the necessary activities.
	Partnership	A Partnership is an agreement of cooperation between two companies.
Financial Aspects	*Cost Structure*	The Cost Structure is a monetary representation of all means employed by the company's activities.
	Revenue Model	The Revenue Model describes how a company gains revenue from a variety of flows.

Table 6 The nine elements of the Business Model Ontology. Part 2 (Osterwalder, 2004)

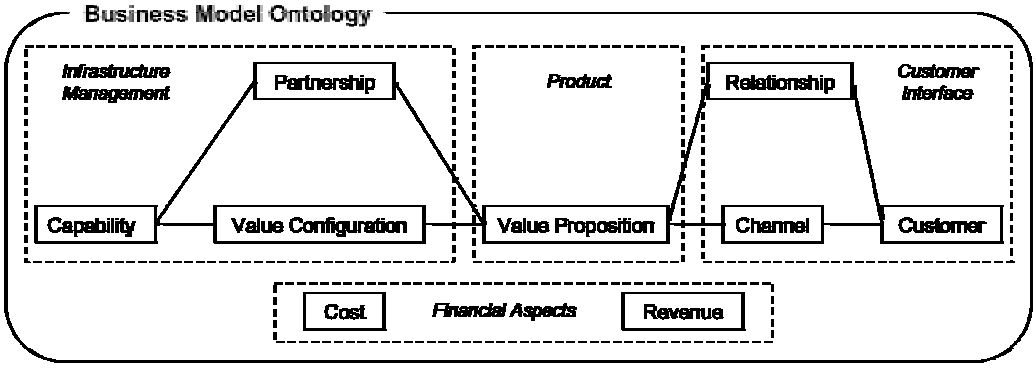

Figure 3 Business Model Ontology. Adapted from (Osterwalder, 2004)

3 Methodology

An overview of the research methodology used is presented in Figure 4.

3.1 Case Study Design

The purpose is to understand the mechanisms that allow firms to extract value from participation in Open Source projects. The forms of our research questions are *what* and *how*. The *what*-question is of exploratory character and most types of research strategies are suitable. The *how*-questions are of more explanatory nature and then case study is one of the better methods. The research questions focus on contemporary events and at the same time, control over behavioral events is not required. This makes case study research a good choice. (Yin, 1994)

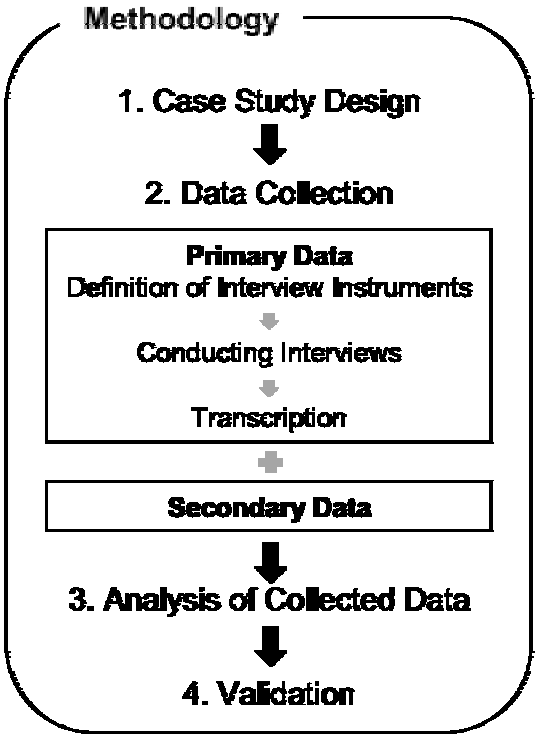

Figure 4 An overview of the research methodology used

The concept of multiple case studies was chosen for the purpose of finding similarities and also to more clearly see what mechanisms, if any, are specific for a given industry or product offering. Together with Elizabeth Hägg, Lund School of Economics, Martin Höst, LTH and Johan Strömhage, Purple Scout AB, four different case studies were chosen. This was done using the following process: first the researchers choose six research objects as a suggestion; then Strömhage had the opportunity to review the suggestions, dismiss any and propose new research objects; finally the other tutors were given a say about how many research objects were feasible. Also a risk analysis was conducted regarding the risk of not acquiring enough contacts on the chosen research objects. The studied objects were selected for their mutual differences: e.g. differences in user groups, size of participating company and difference in industries. The research objects were all selected by project. Firstly, an Open Source project was chosen and then one company that contributed to that same project. These two, taken together, constitutes a single research object in this study.

3.2 Data Collection

According to good practice for case studies, data was collected from multiple sources and viewpoints (Yin, 1994). This is done to enable triangulation and thus attain higher validity (Stake, 1995).

3.2.1 Secondary Data

This was collected from annual reports, industry press, company and project web pages. Information was collected on participating firms, licenses, competition climate, industry developments and the perception of outsiders to the projects. This information gave an understanding of the business and provided a base for further data collection. Also information, vital for business model mapping, regarding firm's product offering, payment models and partners were gathered.

3.2.2 Primary Data

This was collected from semi-structured interviews (Robson, 2002). The interviews were held with people in connection to the project and/or the participating firm.

Definition of Interview Instrument
The research process used a flexible design (Robson, 2002), allowing for changes in data collection during the process. Data collection and analysis thus continued iteratively on each research object until saturation was acquired (Höst & Runeson, 2007).

The researchers studied theory within Open Source and business models and they also studied secondary data about the different projects to be able to ask the right questions. The interview questions were based on a template, see appendix I. The template was used as a safeguard for the researchers not to overlook important research areas. However the differences in organizations and projects required specialized questions for each research object. These were added to the template before each interview.
The interview questions covered the different areas of the business model ontology. This was done to enable an analysis on equal foundations.

Conducting Interviews
Interviews were conducted by loudspeaker telephone or by IP telephone software (Skype), except in two cases were the interview was held in person. The rationale for using telephone interviews was the fact that the interviewees were located in many different countries. Two researchers were involved in interviewing one interviewee. This enabled one interviewer to focus on the questions and talking to the interviewee, while the other could make notes, reflect and come up with other questions. Afterwards they were able to discuss and clarify what the interviewee said to minimize the risk of misunderstandings. In total nine interviews were conducted, see Table 7-10.

Yubico/Yubikey			
Interviewee	*Company/Foundation*	*Position*	*Recorded*
Stina Ehrensvärd	Yubico	CEO	No
Johan Jacobsson	Intraphone IT	Technical Manager	Yes

Table 7 Interviews concerning Yubico

Sun/OpenOffice			
Interviewee	*Company/Foundation*	*Position*	*Recorded*
Per Eriksson	OpenOffice.org	Project Manager	No
Ron Goldman	Sun	Senior Staff Engineer and Open Source Adviser	Yes
Mathias Müller-Prove	Sun	User Experience Architect	Yes

Table 8 Interviews concerning OpenOffice

IBM/Eclipse			
Interviewee	*Company/Foundation*	*Position*	*Recorded*
Andreas Steen	IBM	Chief Technical Manager for Rational, Sweden	No
Donald Smith	Eclipse Foundation	Director of Eco System Development	Yes

Table 9 Interviews concerning Eclipse

Nokia/Symbian			
Interviewee	*Company/Foundation*	*Position*	*Recorded*
Samuli Hänninen	Nokia	Head of Ovi Product Marketing	Yes
David Wood	Symbian Foundation	Catalyst and Futurist	Yes

Table 10 Interviews concerning Symbian

Transcription

The interviewees that allowed audio recording were recorded and afterwards transcribed into text. Which interviews were recorded is shown in Table 7-10. During the other ones extensive notes were taken subsequently and the result was immediately discussed between the two interviewers to avoid misunderstandings.

3.3 Analysis of Collected Data

To aid analysis, statements from the transcribed interviews were summarized and codified into different categories, based on Osterwalder's business model ontology. The same was done with the notes from the interviews that did not allow recording. These codified summaries for a research object were compared between each other for sake of triangulation, and then merged into a unified statement of findings. A model of each research object was then created based on the collected data. The data that did not fit in the business model ontology, for example risks and special strategic concerns, were put in separate categories and were then analyzed using relevant strategy theory.

The analysis compares what theory, secondary data and interviewees have said to get an accurate view.

3.4 Validation

Construct validity is in general problematic in case studies because there is a risk that subjective judgments are made (Yin, 1994). To avoid this the interviews were conducted by two interviewers that were able to discuss the result afterwards. It is also of importance that the interviewees have different positions in the project to avoid a biased picture.

It is difficult to know if the findings are general or specific for the case chosen. This is about the external validity. (Yin, 1994) The research points out that the comparison is between the four projects chosen. In support of generality the cases were chosen to be a heterogenic sample, and should be a good representation of reality. The findings must in any case be taken in context with other literature.

Reliability is about getting the same result if the study were made all over again (Yin, 1994). The interviews are based on a template, see Appendix, to gain higher reliability.

4 The Case of Yubico Regarding Yubikey

Yubico is a small start-up company in the Internet security and authentication industry. The business, started in 2007, is centered on the Yubikey product and the associated authentication solution. The Yubikey is a small identification device that uses the USB-port for power and communication. The solution works as follows: When a user needs to log on to a website, or web-based application, the personal device is plugged into the USB-port of the computer being used. A press of the button on the device generates a one-time code that is filled in the password form. The password is checked with a server and if it matches, the user is authenticated and logged on. The Yubikey solution is comprised of the physical key with embedded software and the server, meaning the software, that is responsible for authentication. (Ehrensvärd, 2009)

Similar solutions are well established on the market; RSA SecurId is one major competitor. The Yubikey solution strives to differentiate in some important areas. First, the device itself offers some advantages: being small, cheap and battery-less. The device uses existing standards for input. It actually emulates a keyboard, and the code it generates is recognized by the application in the same way, as would an input directly from the user. This is simplifying for both developers and users. Besides the technical aspects, Yubico has taken an approach to its software that is deemed radical in the security industry: The software, both the code for the device and the code for the server developed by Yubico, is released as Open Source. The company first adhered to this strategy after an industry pundit misunderstood founder and CEO Stina Ehrensvärd citation that 'the company support Open Standards' to mean that they were going Open Source. After announcing this in the pundit's popular podcast the company decided to try this strategy. (Ehrensvärd, 2009)

The Open Source strategy has fostered the formation of an Open Source community around this specific product. The company has received attention for this strategy, and, despite being a minor business, large corporations are already customers. Ehrensvärd has expansive plans for the business and cites the Open Source strategy as one of their key success factors. (Ehrensvärd, 2009)

4.1 A model of Yubico Regarding Yubikey

4.1.1 Product

Value Proposition

At an aggregate level, Yubico offers similar value as their competitors but with slight innovation and at a lower price. The value proposition can be broken down into individual offerings representing single features or complementary products and services. The identified offerings, structured according to value-addition per product life-cycle stage, are displayed in Table 11. The integration of the product software with the customer's software is simplified since the customer has access to all code. The risk a customer exposes itself to by trusting in the particular security solution is mitigated by the fact the code is open. The customer or other interested parts such as

security experts and industry pundits can inspect the solution and put their trust in the technology rather than the company. The offerings also include a variety of servers, on different platforms and specialized to certain needs that are developed and inspired by the Open Source community around the solution. There is also a flexibility for the customer, made possible by Yubico's revenue model. In comparison with a licensing model, the model offered by Yubico enables customers to experiment with the technology, e.g. installing new servers without having to worry over licenses. They also have increased control of their application management, as they can develop their own environment or benefit by the efforts of others in the community.

Other innovative features, not correlated with the Open Source approach, include the ability to send the small devices by ordinary mail; not needing to replace the battery and the easy one-click solution.

Yubico's Product Offering				
Value creation	*Purchase*	*Use*	*Renewal*	*Retirement*
Easy integration	Small package	Ease of use	-	-
		Support		
		Server software		
		Hosting solution		
		Flexible application management		
		Inspected solution		

Table 11 Yubico's product offering. Entries in bold typeface are benefits from the Open Source strategy

4.1.2 Customer Interface

Customer
The company targets mainly business customers, where there is greater possibility for larger orders. However, the company believes that their revenue model will enable them to also target individual customers. However, offers more directly targeted at individuals have not been developed as of yet. On the topic of geographic segmentation, the company targets developed countries, but see a potential in the developing world, because of the company's lower price structure.

Channel
Besides contact by e-mail, the company's main channel is the webpage as well as the community page, which includes a forum and a Wiki page. This is a collaborative technology, allowing creation and editing of a collection of web pages by registered users. The webpage is run by the company, but the community page is a collaborative effort, involving customers and individuals. These two play different roles, which can be seen if the channel strategy is broken down by the Customer-Buying Cycle (Ives &

Learmonth, 1984), see Table 12. In the awareness stage, the company web page fills a major role, as the community page requires you to actually own a Yubikey to gain access. In the evaluation stage, on the other hand, the community page comes to more use. It enables the potential customer to access the experiences of many others and see the solutions they have tried. There is also the possibility to get support from the community, although in a non-structured manner. The purchase stage is handled via the website, or by e-mail. In the last stage, After Sales, the contact is handled through e-mail. The customer can also get support and software updates from the community page.

Yubico's Customer Channels			
Awareness	*Evaluation*	*Purchase*	*After-sales*
Website	**Community page**	Website	**Community page**
		E-mail	E-mail

Table 12 Yubico's customer channels. Entries in bold typeface are benefits from the Open Source strategy

Relationship

Yubico's current relationship to its customer is focused much on acquiring new customers, which is natural considering that they are a start-up company. The mechanisms that are used are displayed in Table 13, arranged after function. Besides common efforts of personalization and brand building, we consider the customer involvement in the Open Source community. This helps build trust in the technology as well as the company. Being part of the community can mitigate the perceived drawbacks, stemming from Yubico's status as a start-up and the new technology. Other businesses trust in the company will influence newcomers. The overall effect of the Open Source strategy is that Yubico enjoys a closer relationship with their customer. Since the code is open it is easier to engage in a conversation regarding technical issues. In this way the companies can establish a technical relationship that evolves to cover business as well.

Yubico's Customer Channels		
Personalize	*Trust*	*Brand*
Your logo on the Yubikey	**Offer involvement in the community**	The best developer will be a Yubiking
Colours of your choice		

Table 13 Yubico's customer relationship strategies. Entries in bold typeface are benefits from the Open Source strategy

4.1.3 Infrastructure Management

Partnership
Yubico has partnered with Network Marvels, an Indian firm. The partner helps to supply service and support for the Yubikey solution. It also contracts technical consultants for specific development of both software and hardware.

Capability
According to the Business Model Ontology, the company's capability is the result of its combined resources. The identified resources are displayed in Table 14, arranged after particular resource type (human, tangible or intangible). We have modeled the developers contributing to the community as a human resource. Compared to other human resources, such as employees and consultants, these play by different rules: they cannot be managed by normal means but contribute out of own motives. This might motivate the formation of a subcategory to human resources, perhaps crowd or community resource. The community is comprised of mostly technical employees of companies that either evaluate the technology or are already customers. A smaller percentage is individuals, participating outside a business context.

Yubico's Resources		
Human	*Intangible*	*Tangible*
Consultants	Patent	Server park
Community	Brand	
	Software	
	Website	

Table 14 Yubico's resources. Entries in bold typeface are benefits from the Open Source strategy

Other resources, not connected to the Open Source strategy, include technical consultants, patents, brand and a server park. The software is treated somewhat ambiguously by this model. In one sense, the software can be seen as the result of the company's activities, hence a product. In another sense it can be seen as the blueprint of the program, and thus should be modeled much like a patent. Here the latter approach is chosen. What is interesting regarding the software is that because it is Open Source, it is not a unique resource of the company. However Yubico has arguably the greatest use of the resource, as it relates to their particular solution.

Value Configuration
This models the activities required to produce the value offered to the customer. The activities can be performed by Yubico or any of its partners or contractors. Many of the activities rely on one or more of the company's resources. The activities can be arranged by their position in the value chain. Since Yubico both produces hardware and software, two separate value chains are used to model the activities, see Figure 5 for an overview. For hardware the traditional value chain is used (Porter, 1985), see

Figure 6. For software development, the software value chain is used (Spiller & Wichmann, 2002) see Figure 7. The software activities focus on the deployment of the server software and leave out the software on the device itself. This is because the software on the device requires less development and it is therefore of less interest.

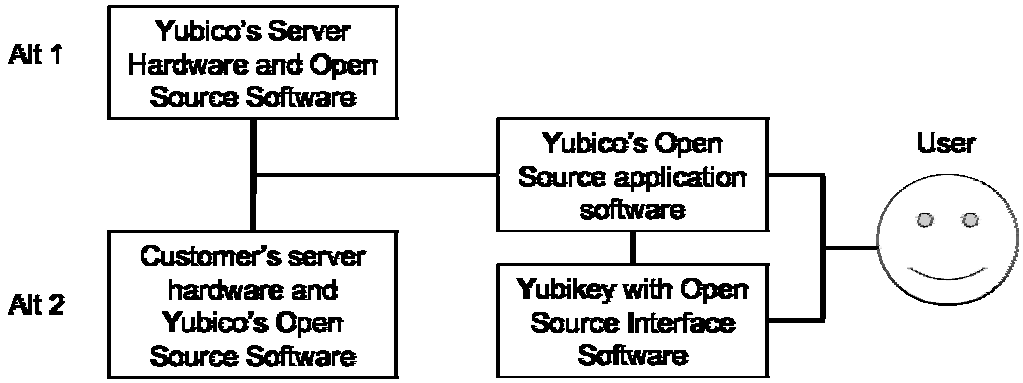

Figure 5 An overview of Yubico's value chains

The value chain relating to the physical device is not much affected by the Open Source strategy. Logistics, production, marketing and service are performed in the traditional manner. It may be interesting to note that the company outsources many of its activities. Functioning much like a virtual business (Malone & Laubacher, 1998), Yubico only has one single employee. What may be affected is marketing and support. Marketing gets a boost from the Open Source strategy since the solution is mentioned in Open Source circles via magazines, conferences and podcasts. There is also some amount of support coming from the community, although this is more relevant for the software.

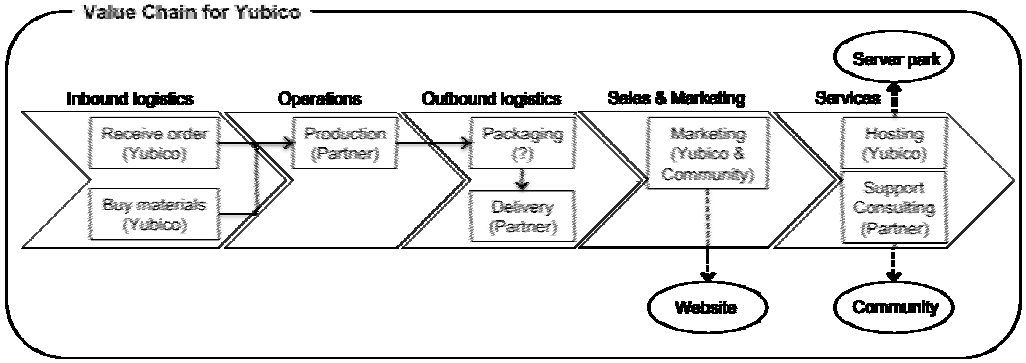

Figure 6 Value chain for Yubico

The software value chain, in comparison, is highly influenced by the Open Source strategy. The three main parts are Development, Sales and Services. The initial development of the server software is done by Yubico, but after that, the community does the majority of the development. This is effective since it significantly lowers Yubico's costs. The code also benefits from new ideas and adaptations to specific environments and platforms that come from community contributions. Software packaging, the second part of Software Development, is not performed by Yubico or

the community. The community lacks expertise, and therefore the customer is sometimes required to configure different versions of software and integrate them themselves.

> *"Yubico's mission is to make Internet identification secure, easy and affordable for everyone. Inspired by IKEA and Amazon.com, we deliver flat packages of do-it-yourself tools."*
>
> (Ehrensvärd, Yubico.com, 2009)

Ehrensvärd comments that they are requiring more responsibility from the customer but at a significantly lower price. The Sales area of the value chain is modified in that the company does not sell the software as such, but rather makes it available. Software marketing is integrated with the marketing of the entire solution. In the Service area of the value chain, Consulting and Support is provided both by Yubico's partner and by the community. Community support cannot be relied on in the same manner as help directly from the company. The community only provides services according to its on decisions, and cannot spend too many hours helping others. Implementation and Training is done by the customer. According to Ehrensvärd, the business model is not fully developed regarding software services. Demands from customers may change future terms of service.

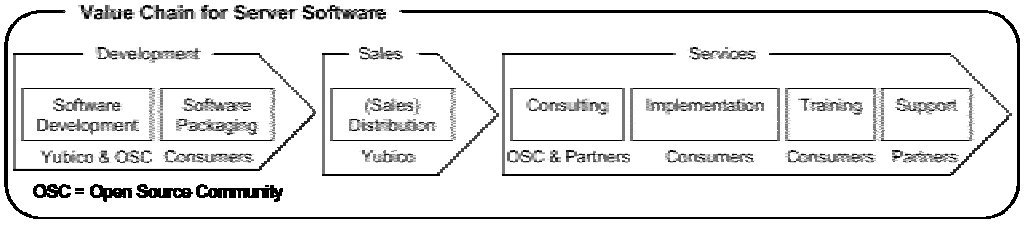

Figure 7 Value chain for server software

4.1.4 Financial Aspects

Revenue

The revenue streams that are part of the revenue model connects to the offerings in the Value Propositions. Because of the Open Source strategy, Yubico cannot sell their software in the same manner as they would with proprietary software. Their main competitors base their revenues on a licensing model, which is not an option for Yubico. Instead their main revenue stream is connected to the hardware, the physical device itself. This is sold with a price that is volume-dependant. The rest of the value propositions, the software notably, is provided free of charge. Compared to a licensing model, this gives the customer more freedom. The company also has a revenue stream connected to the service of server hosting. This is provided via a fixed price subscription.

Cost
The cost model is comprised of a set of accounts, all relating to some specific cost. The main accounts for Yubico relate to the production material cost, manufacturing cost, the salaries of hired technical consultants as well as cost associated with sales and marketing. The model is purely conceptual as it was not possible to obtain quantitative data. Yubico has costs relating to software development but they are relatively low compared to the level of software the company offers. This is a consequence of community code development which the company then can offer to more customers. This is a major benefit of their Open Source strategy.

What is interesting in Yubico's cost model is that the software development cost is minimal, since Open Source development is used.

4.2 Outside the business model

Some of the findings in the Yubico case would not fit in the framework of a business model.

4.2.1 Risks

One is the risk the company incurs by opening the software. Since anyone can study the code, that part of the concept is easily copied, and reverse engineering the physical device is also possible. The company holds a patent for the physical device and so relies on this to protect them from infringement. Whether or not the relatively small company could stand up to a challenge from a larger company remains to be seen.

4.2.2 Strategic Considerations

The effects of lock-in are also not covered satisfactory by the mapping of the business model. A reason for a company to choose Yubico before a competitor may well be in order to mitigate the effects of lock-in inherent in proprietary software. Proprietary software can rely on a certain platform, or certain application management tools. When a customers adapts to these conditions the switching costs become very high, and the customer incurs the risk of being stuck with rising prices.

4.3 Case Summary

In adopting an Open Source strategy, Yubico’s Business Model is affected. See Figure 8 for an overview of the benefits that fits into the Business Model Ontology. The company’s product offerings becomes more valuable to the customer due to higher flexibility and easier integration. The customer can inspect the code to evaluate the security of the solution, and is also able to avoid negative lock-in effects by using Open Source. The company also benefits from viral marketing and being able to form a closer relationship to its customers. Through the community Yubico gets access to more competence. The community contributes primarily with software development but also with technical support to some degree. This reduces the company’s cost structure.

There are also negative effects from the Open Source strategy. The company loses the ability to gain revenue from selling licenses. The company has to manage the relationship to the community, and the software development process is fragmented, negatively affecting software packaging. The company also incurs greater risk of having their solution copied by rivals.

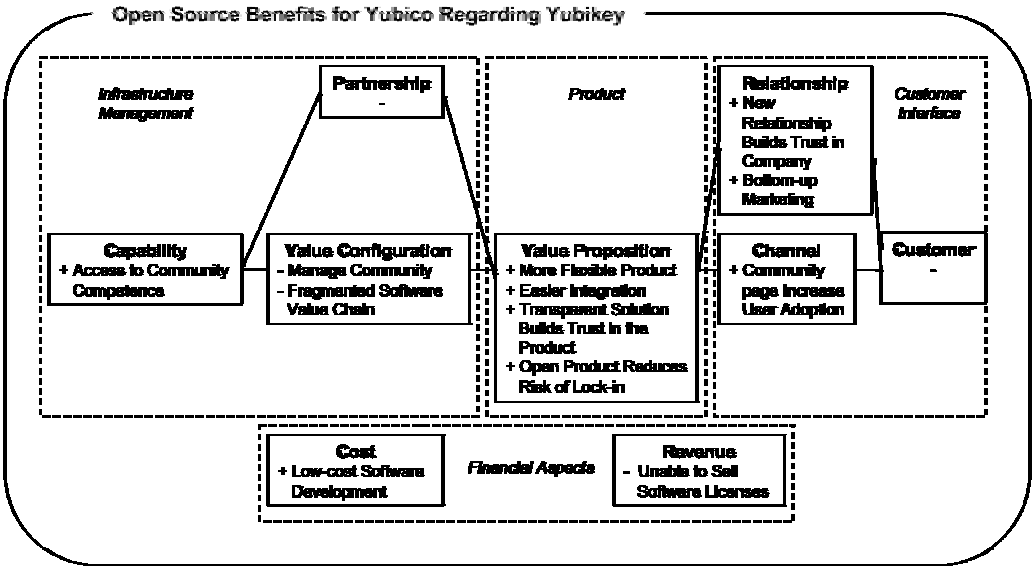

Figure 8 Open Source benefits for Yubico regarding Yubikey

4.4 Discussion

The business model is focused on user adoption. It is easy for potential customers to pick up and play with the technology without entering a licensing process. Integration is also simplified and the customer does not have to worry about the risks of being locked-in. The community provides a loop of positive feedback, creating greater value with every turn. Yubico cannot sell this value directly, but it creates demand for the physical device, which generates revenue for the company. The revenue streams are much smaller per customer compared to its competitors, but this can be compensated by the fact that Yubico has a leaner cost structure and have the potential of creating a large user base. The additional revenue stream connected to services is not large at the moment, but could be an important source of revenue in the future. The demand for service depends on the technical expertise of the user base. Not all customers may want to be responsible for software packaging and implementation.

User adoption is the key to this business model. More users primes the positive feedback loop, creating more value and demand for the physical device sold by Yubico. Because they have non-commodity hardware coupled with the Open Source software, the company is able to capture value created by the community. The company runs the risk of not being able to generate a large enough user base or being copied by a rival company.

5 The Case of IBM Regarding Eclipse

IBM is a global computer technology and IT-consultant company with headquarters in the USA. The company traces it roots to the early 20th century. IBM operates in the information technology industry as well as in the semiconductor industry. The company has a large R&D department, earning 4,186 U.S patents in 2008. Historically the company has been focused on selling hardware and software, but a changing environment during the 90's caused IBM to refocus its business. Divesting its personal computer assets and growing its service organisation, the company shifted its customer target to larger organisations. Today more than 50% of IBM's revenues come from its service offerings. (IBM, 2007)

IBM's Open Source strategy dates back to 1999. It states that the company goal is to support implementations of open standards, through Open Source, to provide choice for the customer. (Capek, Frank, Gerdt, & Shields, 2005)

5.1 Eclipse

IBM started the Eclipse project in November 2001 by donating code for its internally developed IDE, Integrated Development Environment. A consortium of independent software vendors formed a community around the project, collaborating to continuously develop the all-purpose Eclipse IDE. In 2004 IBM formed a not-for-profit organization, dubbed Eclipse Foundation, to steward the project. The foundation was formed to foster a vendor-neutral and transparent community. A board of directors controls the direction of the projects and consists of the foundation's strategic members. The foundation employs a fulltime staff, but does not themselves develop the hosted projects. Nowadays the function of the foundation has grown to include the development of the business ecosystem around Eclipse. (Eclipse Foundation, 2009)

The main application of Eclipse foundation is the original IDE. It started out as a general tools integration platform but has since then evolved to become a rich client platform, RCP. An application developed on the RCP benefits from existing functionality such as infrastructure, GUI and cross-platform support. (Smith, 2009)

The foundation hosts over 100 projects, focused on such diverse products as plug-in modules for Eclipse IDE, tools in the mobile tool chain, platforms for Ajax development and more. The community consists of over 1000 committers working full-time on Eclipse projects, of which nearly everyone is an employee of some member company. On several projects many different companies collaborate to develop the code, but some are completely run by a single member company. The foundation uses the Eclipse Public License, EPL, described on page 13.
(Eclipse Foundation, 2009)

5.2 A Model of IBM Regarding Eclipse

Because of IBM's sheer size, our model of the company will be more general than that of Yubico. For sake of analysis the focus will be on the parts affected by the Open Source strategy, as otherwise the interesting parts would be lost amongst a detailed description of the company.

5.2.1 Product

Value Proposition
IBM product offerings consist of computer hardware, computer software and services. Within computer hardware the company sells server, storage as well as mainframes. It also offers a range of semiconductor products. Having sold of most of its business units that focused on consumer electronics, the remaining hardware focuses on larger-scale computing. The company's software offerings cover such areas as Systems Management (the Tivoli product category), Application Integration (WebSphere), Collaborative Applications (Lotus), Software Development (Rational) and more. The service-part of the organisation, IBM Global Business Services, offers services ranging from pure IT and application management to business consulting and outsourcing. (IBM, 2009) One of the interviewees described IBM's product offering as*: if it plugs in to the wall, IBM can make it work.*

Eclipse software underlies some of IBM's software, for example both Rational and part of the Lotus software. These applications utilize the framework provided by Eclipse to handle application infrastructure and GUI. A positive effect of this is that the GUI concepts used by the Eclipse platform are standardised. This provides a familiar mode of interaction between the user and the program, thus lowering the user's learning curve.

5.2.2 Customer Interface

Customer
IBM's Open Source strategy does not affect these elements of the business model in any profound way. The company targets mainly larger businesses and organisations with complementary offers targeted at the smaller ones.

Channel
IBM has a wide variety of channels to reach their customers, including a dedicated sales force and an extensive webpage. A deeper analysis of these common channels is not of much interest to our analysis. The only way the Open Source strategy affects this is in the addition of the community webpage. Much as in the case of Yubico, this enables a lower-end communication, aimed at the technical departments of other companies. It also gives private individuals the opportunity to access the Open Source technologies developed in the different projects at Eclipse Foundation. The community page works in conjunction with the special relationship between IBM and the outside world, enabled by the Open Source strategy.

Relationship
The Open Source strategy allows for a special relationship between IBM and the outside world. This works both on the level of individuals and organisations. Individuals have the possibility of getting involved in the Open Source projects. However the data suggests that this does not happen very frequently. The major quantity of contributions on Eclipse projects is provided by professionals, appointed by their organisation to work on the project. Instead, the largest impact on individuals lies in the adoption of the software. A person learning to use Eclipse software has crossed a learning barrier. Whether it is just using the Eclipse IDE for developing software or using the platform to create applications, the individual has made an investment in time and energy to learn the workings of that software. The individual takes this preference with them in their professional life, thus affecting the knowledge base of organizations from the bottom-up.

Organizations can also enter in an open-ended technology-focused relationship with IBM. The Open Source license is a commitment from IBM's side, not to withdraw the technology or to charge money for it in the future. Also the by-laws of Eclipse Foundation ensure that IBM has no special position to influence power over the projects. The effect of this is that other actors can begin using the technology with low risk. When the organisations rely more and more on the Open Source technology so increases their need for service and more elaborate features, offered by IBM.

In conclusion, the Open Source relationship between IBM and both individuals and organisations, is focused on customer acquisition. The open technology allows for outsiders to experiment and learn the software, providing positive effects for IBM.

5.2.3 Infrastructure Management

Value Configuration
The activities that are connected to the Open Source strategy are mainly running Open Source projects. IBM itself produces over 40% of all software development within Eclipse Foundation. In several of the projects, IBM is the sole contributor. IBM has relieved itself from directly managing the community, leaving this task to Eclipse Foundation.

Capability
The Open Source strategy affects the resource base in several ways. Since the company has decided to use the Eclipse platform as the basis for many of their developed applications, the technology is present throughout the company. This helps to lower barriers between business units and builds a unified competence base. This could of course have been done without opening the source to the outside of the company, but it is a useful side effect (this notion of open source constrained to within a company is being called "company source").

The projects within Eclipse Foundation that are run by other member organisations produce software resources that are integrated with Eclipse Software. This software is basically free from IBM's point of view, and is produced by companies with other

competencies. There are also several projects where different organisations commit resources to, and thus IBM's human resources are extended to include employees of other companies. A good example is Oracles donation of a complete persistence framework (fundamental database software) to the Eclipse Foundation. This is a large quantity of useful code that is now available to IBM.

Another aspect is the fact that the company has made it easier for themselves to acquire new human resources. Because individuals outside the company have become familiar with the software IBM is using as the base for much of its product, there is a plethora of skilled individuals ready to be hired. Also because the technology is familiar there are more people who want to work for IBM. One of the interviewees stated that there was no need for their department to look for new hires; the hires came looking for them.

Partnership
The Open Source platform facilitates integration between different applications, provided that they both use the Eclipse platform. This makes it easier for IBM to find development partners among smaller companies. Anyone can start developing applications on the platform and their application will be easy to integrate with other applications on the platform. An example is the Black Duck[1] and their software scanning application. IBM has partnered with them, and easily integrated their software in their Rational suite of applications.

5.2.4 Financial Aspects

Revenue
The Eclipse Foundation uses the Eclipse License, a non-reciprocal license. This means that any software released under this license can be used in other products without the need to disclose that software. This is a "business-friendly" license allowing IBM to build products on top of Eclipse software and sell it under a different license. This puts little constraints on the revenue model and so IBM can make money from its software by licensing it. The company is not able to charge money directly for that code which is licensed under the Eclipse license (e.g. the Open Source software), but by adding features on top of that code, revenue may be earned. Thus, IBM earn revenue from value-added and complementary software.

The revenue model for hardware and service is not affected much by the Open Source strategy. Some software can be bundled with hardware and service, and so the Open Source software can boost revenues from hardware and service.

[1] The application from Black Duck actually scans software and finds pieces of Open Source code that has been incorporated in that software. This allows companies to control their codebase, mitigating the risk of inadvertently invoking effects of any reciprocal licenses such as GPL. (Black Duck, 2009)

Cost
IBM's Open Source strategy allows for several ways in which the company lowers its cost structure. Firstly, using the Eclipse platform streamlines software development. One IBM employee estimated that common application development requires 80% of development time for infrastructure and GUI code, leaving only 20% for developing what is specific for the application. Using a well-developed platform this ratio is reversed. Thus IBM can develop applications faster and with lower costs.

The company shares developing costs with other companies, both within and between projects. The products of those projects can be utilized in IBM products or can be utilized in service offering from the company.

There is an enormous benefit to gain from not having to license software you need to build infrastructure for a service customer. Initially the company developed more than 80% of all code within Eclipse Foundation, but now that percentage has decreased to around 40% due to increased contribution from other members. Since the company has a diversified customer offering, there is a high probability that IBM can put a given new piece of code to good use.

IBM does not have to invest heavily in education for its developers. Since the technology is known outside of the company, new hires need not be educated from scratch.

5.3 Outside the Business Model

5.3.1 Risks

There are risks associated with IBM's Open Source strategy. There is a possibility that the development of the platform will go in a direction not favoured by IBM. The company has no special position formally and can therefore not influence more control than any other strategic member of Eclipse Foundation. However IBM is still the largest contributor to Eclipse which, in reality, gives them more control since they can lead more projects. Also, if the projects drift in a new direction it may be caused by influence of the market, which would make the direction a good one.

5.3.2 Strategic Considerations

The effects of IBM's Open Source strategy do not stop at the boundaries of the company. Much of the identified benefits rely on the ability of Open Source to change the dynamics of markets. At the time IBM founded Eclipse Foundation with the donation of a complete IDE, the market was locked up. Microsoft dominated the market with their IDE. One effect of the Eclipse donation was that the market was loosened and IDE software became a commodity (SUN also made an Open Source move with their IDE, Netbeans). This benefits IBM in the long run.

Another effect is that ecology has formed around Eclipse, due to the large user community. Eclipse is in the centre of many new projects, and gains from the

innovative ideas produced there. IBM has a practice of acquiring small companies with novel products and can thus incorporate innovation coming from outside. There is also positive feedback coming from such a user community. The technology attains viral marketing and the complementary products increase the customer's perceived value.

Becoming a standard among users can create a kind of lock-in, solely because those users have become used to that kind of product. The investment needed to learn another application is a disincentive to switch from the already learned application. This is softer than the lock-in attained by purely proprietary software, since the core product is free - both free of charge and free to be modified. But IBM can still stand to benefit since a user groomed on the Eclipse IDE can easily start using their Rational software.

5.4 Case Summary

The Open Source strategy influences IBM's business model in several ways, summarized in Figure 9. It enables a new kind of relationship between the company and outside organisations and individuals, focusing on technology. This helps attracting new customers. The company gains a larger resource base from people participating in the Open Source projects and from code donated to Eclipse Foundation. The company can also benefit from the fact that it is easier to find competent people to hire. IBM's cost structure is streamlined due to shared costs of development between members of Eclipse Foundation. The company also benefits from cheaper in-house development based on the Eclipse platform. The access to code donated or co-developed within the foundation also decreases dependence on third-party code in service offerings, which decreases expenditure on licenses.

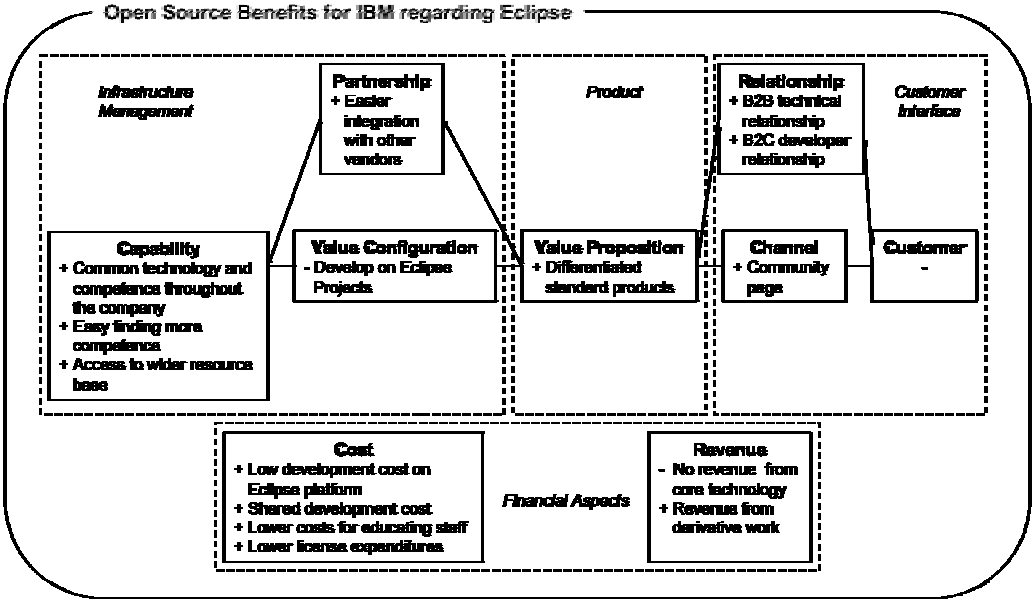

Figure 9 Open Source benefits for IBM regarding Eclipse

The Open Source strategy also influences the market creating strategic advantages for IBM. Users become accustomed to the software, the ecology around Eclipse provides innovation and the Open Source products can shift value in the market by commoditising.

5.5 Discussion

The basis for the advantages that IBM enjoys from its Open Source strategy lies in the shared development cost model of Eclipse together with the business-friendly license. Core technology can be co-developed in a transparent manner, while receiving input and spurring adoption among the broader community. The license model ensures that IBM can develop differentiated applications on top of that core technology or develop applications that are complementary to the Open Source technology. A good example is the Rational software suite, which is built on top of the Eclipse IDE but provides added functionality and more value for the customer. But there are more benefits; as other member organisations donate code and pilot new projects, more useful code is made available to IBM, code the company can use to increase the margins of its service offerings.

The open code also creates a new relationship with the general developer community. Knowledge of IBM technology is spread outside of the company creating a larger impact of that technology, ultimately creating a de facto standard among developers. This lowers IBM’s education cost and increases demand for IBM products.

6 The Case of Nokia Regarding Symbian

Nokia is a multinational communications company with origins in Finland. The company is the world's largest manufacturer of mobile telephone devices with a market share just under 40%. The company's offerings also include network equipment, through its subsidiaries, and services. Nokia dates back to the 19th century, with a background in rubber production. The company became involved in the telecommunications in the 1970s with the conception of a digital telephone switch. In the 1980s the company was involved in the development of the GSM technology, which laid the foundation for the subsequent boom in mobile telephony. (Nokia, 2009)

In recent years the company has tried to refocus their business towards services. In an announcement in May 2008 the company stated that it wants to be seen not simply as a telephone company, but as a player in the field of Internet services. The service department is divided into five divisions: maps, music, games, messaging and media. The Internet services provided by these divisions were relaunched in 2007 under the umbrella name OVI. In 2009, Nokia announced OVI Store, an application marketplace for mobile telephones. (Nokia)

6.1 Symbian

Symbian Operating System, here referred to as "Symbian", is a mobile operating system developed by Symbian Ltd. The company was founded in 1998 as a joint venture between Psion, Nokia, Ericsson, Matsushita and Motorola. The rationale for this was the merging technologies of mobile telephones and PDAs. The advances in technology allowed for a more complex mobile phone, requiring an operating system to handle more advanced applications. The company developed the operating system and sold licenses for its use in mobile telephones. (Symbian Foundation, 2009)

Symbian is the market leader in the market for operating system on smart mobile devices. It consists of over 40 million lines of code, divided into over 100 packages. The complexity of the code reflects the fact that it operates on a wide variety of hardware. The operating system is commonly bundled with the S60 software platform, developed by Nokia. The software platform provides application support on top of the operating system, enabling execution of applications written in various programming languages. (Wood, 2009)

In 2008 Nokia acquired all the shares of Symbian Ltd and announced its intentions to release the code as Open Source. The future Open Source operating system will be bundled with Nokia's S60 platform. The bulk of Symbian Ltd's staff has been transferred to Nokia and the acquired company will cease to exist. Instead Nokia founded Symbian Foundation to steward the code and plan future development. The foundation is made much in the image of Eclipse Foundation but with changes made to reflect the more complex environment of mobile telephony. (Wood, 2009)

At the time of writing, the foundation has been established but the code is still undergoing the process of being readied for publishing. One obstacle that has to be overcome is the problem with third party intellectual property. This regards the fact that not all of the code for Symbian is owned by Symbian Ltd, but is simply licensed from third parties. Either the ownership of these pieces of code has to be transferred or the code has to be replaced. If this cannot be resolved before release date, this part will not be published in the community. The code is scheduled to be published in the second quarter of 2009. During the initial 15 months the code will be available as community source, only accessible for members of the foundation. After that the code will be licensed under the Eclipse Public License. See the license chapter on page 13 for more information about this. (Wood, 2009)

The fact that the company is in the process of realizing their Open Source strategy is complicating the research. As an effect the analysis is more representative of how the business model is designed to function rather than being a representation of present-day facts. This also affects validity in this case.

6.2 Techno-economic Developments in the Mobile Market

The rapid technological development of the mobile market makes for recurrent market disruptions. New developments drastically change what kind of functionality can be harboured in the form of a mobile phone, both in terms of hardware and software. This decennium has seen the advancement of hardware, providing larger screens that can also be touch-sensitive. Hardware capacity has also increased in terms of processing power, battery power and memory capability. Advancements in network technology has resulted in new higher bandwidth communication networks i.e. the 3G network. Wi-Fi technology, initially employed for laptop computers, has moved to mobile phones, allowing Internet access via local networks. The result is an increased range and complexity of applications and services that can be used in the mobile telephone format. Software is becoming increasingly important for the mobile phone and the market for services and application is growing.

At the same time the merging fields of technology allow the entrance of actors traditionally operating in the computer or Internet service industry. Examples of this are Apple's mobile phone, Microsoft mobile version of its popular operating system and Google providing Internet services to mobile phones. In the market for mobile operating systems, Symbian is now competing with Microsoft's Windows Mobile as well as with Google's Open Source operating system, Android. The same companies also provide services on their own and other operating systems. (Google, 2009) (Microsoft, 2009)

To fully understand Nokia's advantages from adhering to an Open Source strategy, the effects thereof should be viewed in the light of these complex market circumstances.

6.3 A Model of Nokia Regarding Symbian

6.3.1 Product

Value Proposition
Nokia delivers value in form of mobile telephones. This increasingly complex product is built in several layers of technology: first hardware, then an operating system, an application platform layer, applications, and finally the services and the service infrastructure. The services require service infrastructure external to the mobile telephone itself. This infrastructure can be located on the Internet, on the telephone network or both. The service infrastructure is, among other things, used to distribute and sell applications to the user.

Nokia value propositions spans the whole range from hardware to services, see Figure 10. The value of the different layers is delivered in different ways. Both the operating system and application platform layer are hidden from the user. They still deliver value to the user: operating system in allowing the use of the hardware capabilities and the application platform in allowing many different applications to be executed on the handset. However, from the user's point of view the layers are not a differentiating factor in itself. The physical handset on the other hand, as well as the applications and services provided, gets more interaction and is more visible to the user. Their value is delivered closer to the user.

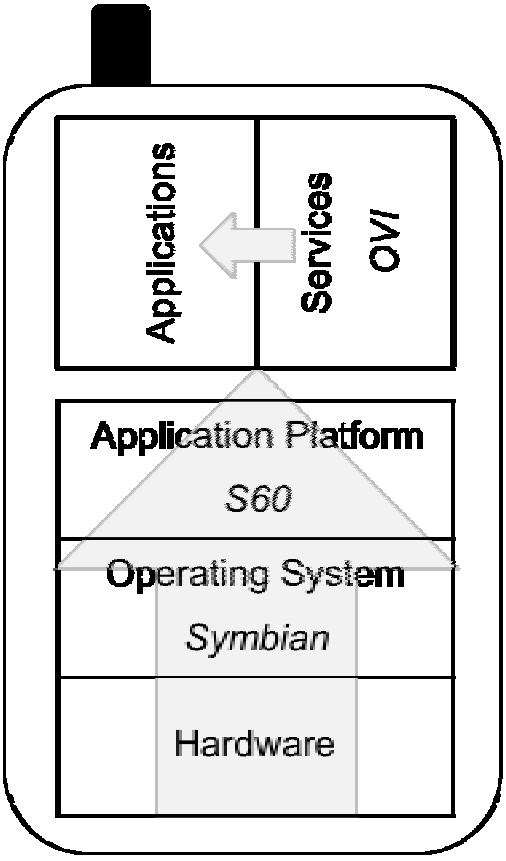

Figure 10 Nokia's product layers

From the application vendor's perspective, however, the operating system and application platform are more important. Here the other applications and services are merely complementary products. Also the service infrastructure plays an important role as it determines how the applications are sold and delivered to the handset.

The benefits of making the Symbian operating system and the application platform Open Source is that making it open and positioning it as a standard increases the innovative power of the combined platform. This will deliver increased value to both users and application vendors – indirectly to the former, directly to the latter. The application vendors are also offered more influence over the development process of the platform. Positioning as a standard increases the platform's value for the

application vendors due to positive network effects and increased user base.

Opening the platform means relinquishing the ability to differentiate from other actors on this layer. However this is no problem since these layers are not the ones most visible to the user. Nokia believes they will be able to differentiate themselves adequately towards users through handset design and services. Towards the application vendors the differentiation lies in the nature of service infrastructure and the size of the user base.

6.3.2 Customer Interface

Customer
Nokia serves mainly three types of customers. On one hand, the mobile telephone users and the operators. The handsets are either sold directly to the users through retailers or through the operators to the user, in that case bundled with subscriptions. In both ways the user is in focus as they are driving demand.

The other aspect is that of the application vendors. This is a newer segment of customers where the relationship approaches partnership. The vendors sell applications to the users through the marketplace provided by Nokia (the Ovi Store) and in doing so are required to share the revenues with Nokia. The vendors are commonly small-to-medium-sized businesses whereas the consumers are private persons.
The Open Source strategy does not infer any direct benefits in this area.

Channel
Nokia supports channels through which they work to establish a closer relationship with users and application vendors. The My Nokia website is focused on user relationship and the Forum Nokia website focus on developers. The newer OVI Store functions as a marketplace, connecting users with application vendors. The application store works on Nokia handsets running Symbian, but it is yet uncertain whether it will also be available on Symbian handsets from other manufacturers than Nokia. Nokia is focusing on deploying the store on the Symbian platform, but will consider other platforms in the future.

In addition to these channels, Nokia will gain a channel to the community, in form of Symbian's community page. Since this channel has not been established at the time of writing the effects cannot be righty evaluated. However, Nokia's intentions are to draw developers even closer to themselves.

Other than the Symbian community page, the Open Source strategy does not directly influence Nokia's channels. However the Ovi Store works in conjunction with the Open Source operating system to deliver value to application vendors. This in turn adds to the value delivered to users through means of the products of the application vendors. The combination of an open platform, generating innovation and flexibility, and a direct channel to users is valuable to developers.

Relationship
In conjunction with the Open Source community channel, Nokia can work to create a closer relationship to the application vendors. The relationship to the other customers, the handset users, is not affected by the Open Source strategy.

6.3.3 Infrastructure Management

Capability
Nokia has resources connected to developing and manufacturing handsets. This includes factories, hardware design competence, patents and logistical systems. To deploy service offers the company has also acquired a service infrastructure and related development competence. However these resources are not influenced by the Open Source strategy.

Through acquiring Symbian Ltd, Nokia gained access to its software development competence. The majority of Symbian developers are now part of Nokia's development team. Nokia had previously competence mostly about the internally developed S60 application platform but now the two different development teams have merged. This competence is an investment from Nokia's side in the Symbian platform. If one considers that the Open Source strategy is a supportive move for the platform it is clear that the strategy safeguards the relevance of Nokia's competence. If the platform's popularity should diminish in favour of another company's operating system, much of Nokia's software competence would be of less use.

The Open Source strategy allows Nokia to gain access to competence outside of the company. The Open Source operating system can now, for instance, be modified to specific devices or to support functionality whose development requires competence not found in Nokia. Symbian Foundation has already attracted many members, but it is too soon to say how many of these will actively contribute in developing the code.

Value Configuration
Nokia has many activities related to the handset's value chain, however these are not affected by the Open Source strategy. Also the development and hosting of services are separated from the inner workings of the platform development and is not affected.

Since the formation of Symbian Foundation the foundation's boards will handle the platform's planning and decision-making. However Nokia employees will perform the majority of software development since the company holds much of the competence required.

Nokia must now manage its relationship with Symbian Foundation in order to maximize its influence all the while they have to support the foundation's non-biased modus operandi.

Partnership
One of the effects of using Open Source is easier and deeper integration of software. Developers other than Nokia can more effectively develop applications and services for the platform aiding the formation of partnership.

6.3.4 Financial aspects

Revenue
The biggest change of the Open Source strategy is that the company will no longer gain license revenue from the S60 application platform or the Symbian operating system. To make up for this revenue loss the company hopes to sell more handsets and also gain revenue from services and applications.

Cost
The company will share the development costs of the Symbian operating system with the other members of Symbian Foundation. In the beginning, however, Nokia will probably still be the major developer.

6.4 Outside the Business Model

6.4.1 Risks

There are risks associated with moving to Open Source. Publishing the code will reveal all the inner workings of the software, possibly revealing security flaws in the program. These could be exploited and cause harm to users and ultimately to Symbian's reputation. The code is being studied for such security gaps prior to release, but considering the size of the code base the risk is hard to eliminate completely. That is part of the reason that only the community, that is members of Symbian Foundation, will have access to all the code during the first 15 months of its release. The reasoning goes that the members can inspect the code and hopefully find more possible security flaws before the broad public gain access.

6.4.2 Strategic Considerations

One cannot fully understand Nokia's Open Source strategy without taking into consideration the competitive landscape. Symbian competes mainly against operating systems from Microsoft (Windows Mobile) and from Google (Android). Microsoft is using a traditional license model for its operating system, whereas Google provides theirs as Open Source. Symbian still holds the largest market share but faces competition from two sides. Microsoft is a software giant with large amount of resources available to be poured into software development and market penetration. Also they have a history of successful domination in the PC operating system market. On the other hand Google challenge with an Open Source operating system, provided free of charge and open for development and study.

A mobile operating system enjoys positive network effects that come with increased user base. More users and developers increase the value for all other users and developers. The case of mobile operating systems can thus be viewed as a classic

platform war. However while the battle rages to position different operating systems as the leader, there is also a development towards commoditization. When there are viable operating systems available without license costs the possibility to charge money for any operating system is reduced. Also when the functionality of the operating systems grows to cover support for most common technologies, they become more and more similar. The market for mobile operating systems is thus undergoing a platform war for commodities.

If it becomes harder and harder to gain revenue from operating systems, why bother fight a war over them? One reason is that once an operating system establishes leadership and gains strong enough lock-in effects, the price can be increased. Also, leadership in the world market creates good Economy of Scale, so that even a low price can be profitable. In the case of Open Source operating systems however, the company needs to retain the copyright of the entire operating system to be able to relicense it.

In the case of Nokia and Symbian, Nokia has much to lose if Symbian ends up as a loser in the platform war. The company has invested in competence for the platform - software and hardware are all compatible with Symbian, as well as the new service infrastructure, Ovi Store. Therefore Nokia stands to gain if Symbian is reaffirmed as the standard platform. Making the operating system Open Source is then a good move, considering that it lowers the price to zero, in line with the commoditizing of operating system, and considering that it increases the innovating power of the platform. The other way to go would have been to fight the platform war with closed code and company resources alone.

> *"If we hadn't done that I guess the innovative capability of the platform would have started to deteriorate, but now it's making a very strong showing again."*
>
> (Wood, 2009)

Since Nokia started a foundation, the code is safeguarded from later relicensing to a proprietary license. This creates a good incentive for other companies to trust in the platform. This primes the positive feedback-loop and promotes the software's development and proliferation.

6.5 Case Summary

Nokia's Open Source strategy benefits their Value Proposition by providing a more innovative platform to users, and a more open platform to the application vendors. Nokia also gains a closer relationship with application developers through the new channel of the Symbian community. The company acquires access to competence from outside the company that can benefit the development of the Symbian platform and share the development costs. Nokia also benefits from the continued dominance of the Symbian platform in the market for mobile operating system, a scenario that is promoted by the Open Source strategy.

Negative effects include not being able to sell licenses for the software and having to manage the community and balancing the fine line between neutrality and influence. The company loses control over the S60 application platform. Nokia also runs the risk of getting badwill from any security gaps that are revealed in the code. See Figure 11 for an overview of the benefits that fits into the Business Model Ontology.

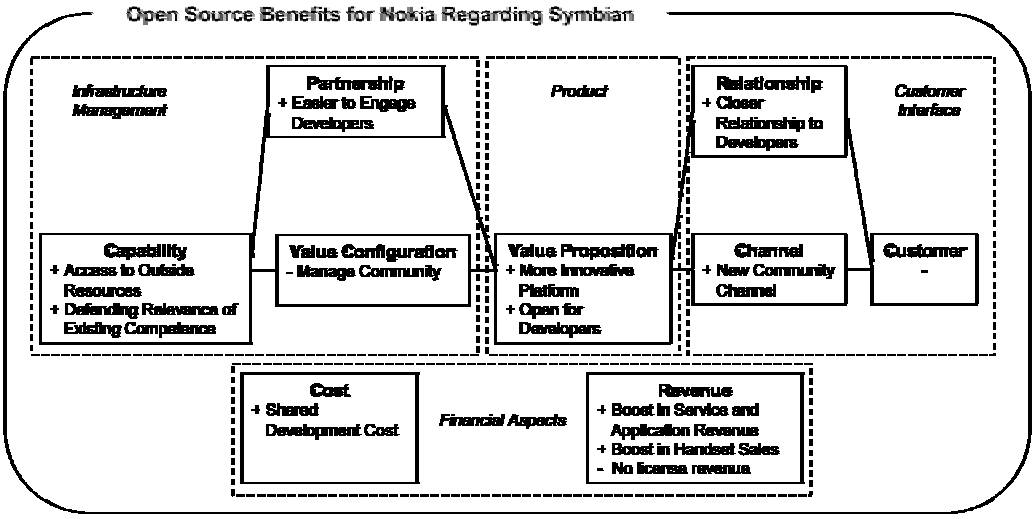

Figure 11 Open Source benefits for Nokia regarding Symbian

6.6 Discussion

The most important aspect for Nokia is the continuation of their platform as a market leader in the future. As operating system and mobile telephones evolve, the focus has shifted, in the opinion of the authors, upwards toward services and applications, all the while handset design remains important to users. Nokia moves the company to follow this evolution by trying to promote Symbian as a platform leader while differentiating on services, applications through third-party developers, and traditional handset design. It remains to be seen how much the revenues from services and application will grow and how the platform war will play out.

7 The Case of Sun and OpenOffice.org

7.1 Sun

Sun Microsystems is a multinational vendor of computer hardware, software and IT services. The company has traditionally gained most of its revenues from hardware sales. Sun specialises in advanced server systems built around their own architecture and the Solaris operating system. The company also develops and sells web infrastructure software, developer tools and database software. The popular programming language Java originated from Sun. (Sun Microsystems, 2009)

7.2 OpenOffice.org

OpenOffice.org is an Open Source office application suite and also the name of the Open Source project that aims to develop the product. The project was started in 2000 by Sun Microsystems and they continue to stand as the project's main sponsor. In 1999 Sun acquired a German software company, StarDivision, that developed an office suite called StarOffice and it was the donation of their code base that became OpenOffice.org. (OpenOffice.org, 2009)

The OpenOffice.org office suit provides functionality similar to its main competitor Microsoft Office. The suite includes application for word processing, spreadsheets, presentations and databases. The GUI, Graphical User Interface, is explicitly designed to be similar to that of Microsoft's products to ease the transition for users. The applications use Open Document formats, but also support common proprietary formats. The code for OpenOffice.org is licensed under the LGPL license described under the license chapter on page 13. (Müller-Prove, 2009)

The community around the OpenOffice.org project consists of three groups: Sun employees, developers from other software companies and users. The major part of actual code development is done by Sun, 80-90 % according to Mathias Müller-Prove at Sun. The large community of users assist with other non-coding functions, i.e. language localisation, marketing and design of document and sheet templates. The contributors employed by other companies all have their expertise in certain areas where their employer would like extra functionality to be implemented. However these contributors are a minority in the community. (Müller-Prove, 2009)

7.3 StarOffice

StarOffice is Sun's proprietary version of OpenOffice.org. The product is built on Open Source software but is not itself Open Source. Sun sells and distributes StarOffice as an alternative to OpenOffice.org. There are actually no major differences between the two. StarOffice is built directly on the OpenOffice.org codebase. *Internally it's just a switch,* according to one Sun developer. StarOffice adds a different, proprietary dictionary and also includes tools to aid migration from Microsoft Office. Sun also provide support for StarOffice. (Müller-Prove, 2009)

7.4 A Model of Sun Regarding OpenOffice.org

Sun is a large company with a varied product portfolio. What complicates the case is that the company is involved in many Open Source projects, including such projects as Solaris, the Java platform, the MySQL database and other. These products have a different aim than the OpenOffice.org project. *They are more hardcore*, as one Sun software developer puts it, focusing more directly on software developers and IT-infrastructure. Sun's broad Open Source strategy is connected to all of these projects. This thesis aim, however, is to consider the OpenOffice.org project separate from these other projects even though they also offer an intriguing area of study. Because these other projects centers on products in Sun's portfolio they will figure as a backdrop in this analysis, but not as representative of the studied Open Source strategy.

7.4.1 Product

Value Proposition
Sun's main business area is large installations of servers and storage. In addition to hardware they offer software built to enhance the productivity and application of this hardware. For example, the servers are distributed with either the Linux or Solaris operating system and there are additional offerings aimed at providing platforms for running applications or database software for handling storage. Sun also offer solutions for *desktop virtualization*. This is a technology that allows a server to emulate the functionality of several individual desktop computers. Instead of running several physical machines, the computations and storages are run central on a server, connected to terminals that simply act as a front for the server. These terminals are called thin clients. In this way Sun's servers compete in markets that are connected to desktop computing.

OpenOffice.org offers common desktop productivity on a range of platforms. It complements the other offerings in that one can run a Sun server with Linux or Solaris and still get the standard office applications that one would have one a machine running a Microsoft operating system. This is important for many of Sun's hardware customers.

The application suite benefits from the Open Source approach in that the development is done in close connection to the user community. The community is directly involved in feature request and feedback. This ensures the user-centric development of the software. Also the world-wide community assists in localisation efforts, with the effect that OpenOffice.org is available in many languages, even some very small ones.

OpenOffice.org also benefits from the support of open document formats. This could be supported even without the Open Source strategy but the influence of the open community ensures the support for open standards. Also the customer is safeguarded from many negative lock-in effects. If Sun would increase the price of the value-added StarOffice, there is always the possibility for the customer to revert to the pure Open Source version.

StarOffice, the proprietary version of OpenOffice.org, offers similar value since it is in essence the same software. There are minor differences but these do not offer much in additional value. However a proprietary version offers more trust than its pure Open Source counterpart. When customers pay a front-up price for a product they feel more secure in that there is someone to turn to if problems arise. Some customer also feel more secure in that they believe that since the product generate direct income for Sun they can rely that the company will continue the development. Sun also offer support for StarOffice but not for OpenOffice.org. The customers has a choice to either opt for the freely available OpenOffice.org and care for support themselves, or choose and pay for the proprietary StarOffice which offers more in way of security.

7.4.2 Customer Interface

Customer
One important customer group, regarding OpenOffice.org, is the company's hardware customers. They need the OpenOffice.org software to complete the other offerings from the company. However, OpenOffice.org's largest customer groups lie elsewhere. There are many customers in governments, education and non-profit organisations. The product also enjoys much support from emerging countries, due to strong localisation efforts and the free distribution.

Channel
The Open Source strategy provides Sun with the project's community page, which acts as a channel to the user community. The company uses this channel to aid in the relationship building with the community members.

Relationship
One of the major benefits Sun enjoys from the OpenOffice.org project is in branding. The product is distributed to millions of people world-wide including schools and universities. The OpenOffice.org product comes with Sun's logo, and so does the website. This builds Sun's brand, connecting it to Open Source efforts and collaboration. One OpenOffice.org developer says:

> "*I really have to compare this one to the yellow boots of Konrad Lorentz. […] [The biologist] who grew the little ducks and the first thing they saw was his yellow boots and so they followed his boots. And so it is important for every company to really get their brand or their product in the face of young people, of pupils or students.*"
> (Müller-Prove, 2009)

The attention Sun receives from OpenOffice.org also raises awareness of the company's other Open Source projects, drawing more people to the company's technologies.

The large community also assists in marketing the OpenOffice.org product. The message is spread by viral campaigning and so Sun does not have to fund major marketing campaigns.

7.4.3 Infrastructure Management

Capability
Using the Open Source strategy Sun gains access to a user community. In the case of OpenOffice.org these community members do not possess the competence to contribute with actual software development, instead they contribute by performing other tasks. Feedback on features and feature requests are performed by community members, bringing the user perspective into the design process. Since the community is distributed worldwide the project has access to extensive language and culture competence. The community also contributes with templates for documents and spreadsheets, as well as extensive documentation on features and functionality. There is an ongoing process in the OpenOffice.org project to better tap into the community's potential.

Value Configuration
Sun does the majority of development on OpenOffice.org and has several developers dedicated fulltime to the project. Apart from development, the company also manages the community and hosts the community page.

Partnership
The Open Source strategy regarding OpenOffice.org does not convey any real benefit in this area. There is little need to integrate the code with other software and the user community consists by large of non-developers.

7.4.4 Financial Aspects

Revenue
Sun gets revenue from the OpenOffice.org project by selling licenses and service for StarOffice. Since the product complement other offers, revenues are also generated indirectly by enabling other sales.

Cost
The Open Source strategy allows community cooperation and so the cost for those tasks is decreased. The tasks do not include software development and the costs there have to be covered almost completely by Sun. In addition many of the community's effort must be managed and coordinated for it to be purposeful. In the case of OpenOffice.org cost reduction may not be one of the main benefits. One interviewee stated:

> "*Any company that is putting it in and making an open source project has 120 % for development. They've got 100 % for doing the development plus they've got another 20-25 % for dealing with the community. […] What you get for the extra 25 % is a product that's actually going to succeed. It's going to have the best features in it, meet the customer needs.*"
>
> (Goldman, 2009)

7.5 Outside the Model

7.5.1 Risks

Sun does not perceive that there is any major risk with their Open Source strategy. The claim is that since Sun controls the majority of competence on software development on the project, there is little risk of the project slipping out of their control. Any other company can take the source code and redistribute it with another name, but there would be little reasons for customers to switch. Sun is also strongly associated with the project. Other companies are using OpenOffice.org internally, even modified versions, but such internal use is not considered as a drawback by Sun.

7.5.2 Strategic Considerations

A natural assumption in the case of OpenOffice.org and Sun, is that the project is especially aimed at attacking the market position of its largest competitor, Microsoft Office. Since the two companies are also in competition in the server market, it could be reasoned that hurting ones business in one market could influence matters in the other. However, in this study this factor was not found to be of major importance for Sun.

7.6 Case Summary

Sun benefits from the Open Source strategy because the value of the software is increased due to the developmental model. The OpenOffice.org offering complements the company's other and more revenue-critical offerings, ultimately motivating the project. The fact that it is licensed as Open Source is also valuable to some customers. Sun benefits from the branding effects, building awareness and connection Sun with Open Source projects. The company also benefits from quality feedback and language localisation work from the community. The revenue comes from the proprietary version, StarOffice, but it is not considered as a large source of revenue by the company's standards. Reductions in cost due to the Open Source strategy is also not a big factor since most of the development is done by Sun. See Figure 12 for an overview of the benefits that fits into the Business Model Ontology.

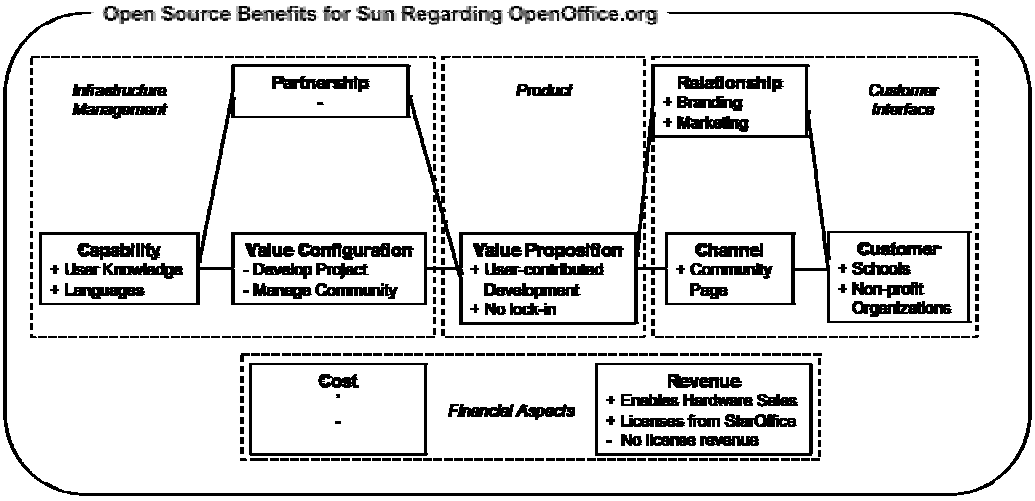

Figure 12 Open Source benefits for Sun regarding OpenOffice.org

7.7 Discussion

OpenOffice.org is an interesting case. Sun has many other Open Source projects aimed at developers where the connection between the projects and the company's complementary products is much clearer. OpenOffice.org is perceived as valuable for a subset of the company's customers and so it acts as a complementary product. The choice to develop it as an Open Source project does not change the relationship with their main customer segment. Changes are more evident in the manner in which the product is developed. The fact that there are some of Sun's customers that are interested in the product is dwarfed by the projects broad public appeal. Since the market for Office Application Suites have been largely locked-up to Microsoft's products, a viable Open Source alternative can be seen as an altruistic act, generating value free to users worldwide. This is of course good branding indeed for Sun, and also the product's success in schools and universities, possibly alongside server installation from Sun, is deemed to wean students to the company's products.

The project has a large community of non-developers, also an interesting property. This means that the company does not receive any larger contributions or innovations in form of code. Instead contributions come in form of bug filing, feedback, feature request and complementary functionality not requiring coding skills. The company vouches for its development model, attributing success to the acting involvement of the multitude of users. One can question the business model of the project. Whether or not it is valuable to the company depends on many factors: how many customers are paying for StarOffice, how many hardware customers requiringing office software to invest in a solution from Sun, and finally how much the final product is made more valuable thanks to the developmental model. However, the answer to this is not attainable by this study.

8 Comparative Analysis and Discussion

8.1 Value Proposition

The Open Source strategy of the researched companies all provided some benefits to their Value Propositions. In both IBM and Nokia their ability to provide products that serves as platforms was increased. In Sun's case the product was perceived to be of better quality due to the user-involvement in the Open Source project. Yubico's product enjoyed a variety of benefits: easier integration, higher security, richer variation and higher flexibility. There are differences among the cases and this thesis proposes that these can be explained by the nature of the product and the way it is utilized. In both IBM's and Nokia's case they seek to develop a platform. The Open Source strategy enables them to collaborate with other actors and simultaneously promote adoption. OpenOffice.org's userbase is mostly non-technical users and so they rarely have the need to look at the code and are unable to contribute to the code. However, the community is involved in the development process, which results in a better product. In Yubico's case the customers are highly technically skilled and need to integrate their product with their own system. The software must work in varied environments and security is a major issue. In this case there are more benefits in using Open Source since the license allows the customers to manipulate the code after their own needs and integrate it with less effort. The access to the source code is more important for Yubico's customers than it is for Sun's customers.

8.2 Customer

Open Source strategy can affect what kind of customer segments can be targeted. Yubico has a better position to target emerging countries because the solution is much cheaper than its competitors. Sun's OpenOffice.org has the same advantage as it is free and can easily be downloaded from the Internet. It's also has an advantage in reaching non-profit organizations and schools.

8.3 Channel

All participating companies grant access to a community page, which is a channel for the community. In cases where community members are also the customers, the channel enables the company to develop customer relationships.

8.4 Relationship

There are several different benefits in this area. Both Sun's and Yubico‘s Open Source participation strengthens their brand. Yubico also benefits from the possibility for new customers to try out and modify the code. In IBM's case the company has found a new way to build customer relationships. Companies can interact with each other on a technical basis, bypassing marketing departments. This can be subtle, but a shifting relationship between vendor and customer is important. Attracting and retaining customers, as well as add-on selling, are vital functions for a company. If using Open

Source can draw the customer closer by enabling technical cooperation over the common code base, then this is potentially very valuable for a company.

8.5 Capability

All researched companies enjoy a wider resource base as an effect of their Open Source strategies. IBM benefits since other corporations are involved in Eclipse projects, bringing new competence into the mix. Nokia and Symbian expect contributions from persons competent on varying types of devices. OpenOffice.org's community holds a smaller proportion technical expertise but contributes user-side knowledge and language skills to the project. Yubico has their software available on many platforms and in many languages, due to competence from their community.

There is a potential for greater innovation here. According to the theories, companies should reach outside of their company to tap into knowledge and ideas that is there. Open Source seems to be one manner to achieve this. Yubico gets new ideas for their server software from community members and customers that bring their own perspective to the mix. In IBM's case we found that they did not view upstart companies that used their Open Source technology as a threat. They are, after all, frequently acquiring smaller companies thus incorporating outside ideas.

8.6 Value Configuration

In all cases we found that the company involved in the Open Source project committed substantial development resources. They were also considered to be the project's driver, even when the projects had other formal decision structures. It seems that the largest committer gains more informal, or technical control of the project, whereas political governance structures can inhibit control over more high-end development decision.

In all cases the companies managed the relationship to the community. In accordance with other research this implies that having a suitable relationship to the community is a major component in the possibility for increased value.

8.7 Partnership

In two cases, IBM and Nokia, we found benefits in forming partnerships. This is perhaps correlated with the fact that both companies' projects are developing platform products. These products are designed to interact with many applications and are often packaged with other layers and software bundles. Thus there is a greater need to form partnership over the code than with Open Source software that is a stand-alone product. OpenOffice.org, for example, does not need to interact with much other software and so partnering is not a big topic. In conclusion, Open Source can help forming partnership over certain software, but it depends on how much the software interacts with other software.

8.8 Cost

Cost reduction was found in two cases: IBM and Yubico. Nokia expects cost reduction to occur in the future but experiences no benefit at the time of research. Sun is not deemed to benefit from any substantial cost reduction, perhaps since their community mostly consists of non-developers. In Yubico's case the internal development resources are greatly dwarfed by the development capabilities of their customer. Since the customers are also part of the community this ratio in size could influence the potential for cost reduction. In IBM's case the community consists largely of companies that are Eclipse members. These can be partners, customers or competitors. However it seems that even among this heterogenic community there is potential for sharing development costs.

8.9 Revenue

Among all studied companies the possibility to gain revenue from licenses was lost with the Open Source strategy. To gain revenue, the companies rely on complementary products. Yubico sells hardware coupled to the software; Nokia sells both handsets and services, as well as gaining transaction fees from applications sold through their marketplace; IBM sells proprietary software with added value or delivers service packages utilizing Open Source software. Sun has a more indirect manner of generating revenue. For them the software acts more as an enabler, increasing sales of related hardware. There is also some revenue gained from selling the proprietary version, StarOffice.

Many of the benefits in other areas of the business model also translate into increased revenue, but in a more indirect manner. Better marketing and branding means higher market share and more revenue; forming more partnerships allows reaching more markets and customers; getting access to a larger competence base means more innovation, new product offerings and more revenue. The effects of these are more difficult to measure since the causality is less obvious.

8.10 Risks

In all cases, except Sun, we found risks associated with the Open Source strategy. Yubico incurs greater risk of having their solution copied; IBM have the risk that the Eclipse platform develops in an unfavorable direction and Nokia is risking exposing security flaws in the software. The risks found were not of major magnitude in any of the cases. Also, there were no risks associated with legal incidents, implying that the companies have a good grasp of the judicial implications of the licenses they are using.

9 Conclusions

In the studied cases we found that the companies involved in Open Source projects captured value from complementary offerings. These complementary offerings included hardware, software and service offerings. The studied companies also gained other benefits that influence revenue streams and cost structure. The extra benefits found in all cases were viral marketing, closer customer relationship and access to outside competence. In two of the cases we also found benefits to forming partnership, and in one case there were benefits to the recruitment processes. In two cases benefits to the company brand were found. In conclusion, the studied companies all capture value primary from complementary offers, but also capture secondary value in form of benefits to other parts of the business model.

In all cases but one we found risks associated with the Open Source participation. These were: risk of exposing security flaws, risk of being copied by competitors, risk that the Open Source project develops in an unfavorable direction. In all cases the risk were found to be of a smaller magnitude.

Context and Further Research
The findings of this study confirm one previous study (West, 2007), but also widen the perspective with the analysis of secondary benefits. The effects and prevalence of such secondary benefits is an interesting subject that could be studied further. In the authors' opinion the shift in relationship between vendor and customer could have significant implications for businesses. The findings also provides support for the Open Innovation Model, in providing a picture for how businesses can benefit from opening up internal work and cooperate with the outside world.

10 Management Open Source Checklist

In this section we propose a management tool, a checklist for evaluating participation in Open Source projects. Based on the findings in this case study as well as on previous research this tool will provide a guide for managers that are considering an active Open Source participation. It proposes to assess the possible benefits, as well as pitfalls and barriers, from a business perspective. It is the authors' hope that the validity of this tool will be tested in future research.

Who will be a part of the Open Source community?

Private citizens:

- Is the product useful by itself?
- Is it cutting-edge?
- Is it exciting?
- Does the product stand out among other projects?

  If the product is not found to be interesting enough it will not build enough traction to start the positive feedback-loop.

Organizations, partners:

- Does the license permit partners to build other products around it?
- Is the technology useful for other organizations?
- Will other organizations trust that you will not control the code too much?

  A common solution to build trust among other organizations is to entrust the governance of the project to a neutral foundation.

Customers:

- Do customers have an interest in looking at the code?
- Do customers have a need to modify or expand the code?
- Do customers have the ability to do this?

  Customers have a self-interest in committing to the project because the modifications they need gets incorporated in the base code and so gets supported even in newer versions.

How will you create value for your customers?

Do they need access to the code:

- Do they need to adapt it?
- Do they need flexibility?
- Do they need to integrate the code?
- Do they need to inspect the code?

To what extent do they want to avoid being locked-in?

Will you get a large enough community to provide innovative power? *(Consider the answers to previous section)*

A large community provides value to the product in general, but there might be value inherent in licensing as Open Source. This depends on the nature of the product and the customer, whether they need access to the code.

How will you capture value?

Do you have complementary products?

- Hardware
- Software
  - Premium versions
  - Plug-ins
  - Managing applications
  - Applications one layer up
- Services
  - Internet services
  - Consulting
  - Support

How tightly are they connected?

- Do they require one another to function properly?
- If not, how often will the customer need the complementary product? How many customers need it? How much?

What are the margins of the complementary products?

How much revenue will you forfeit?

- Is the code a differentiating feature in your offering?
- Can it be sold separately?

Some code is not suitable to be developed or released as Open Source. If the code provides differentiating value to your customers and if they are willing to pay for it explicitly you should not use Open Source.

How will it affect your costs?

How large and how technically competent is your community?

Even if you have a large and competent community, remember that cutting cost is not guaranteed. It takes efforts to integrate community efforts and manage the community.

Are your customers part of the community? If so, do they have technical competence and interest in modifying the code?

Do you have large developing partners that can also gain benefits from the software?

Can you gain additional benefits?

Will the Open Source participation support your brand?

Can you market other offerings through the Open Source project?

Can your Open Source project connect you to developers? Are your offerings targeting developers?

Are your customers involved in the project?
If so you can develop a new relationship with your customers with potential for add-on selling. You can also acquire new customers in this way.

Is there potential for forming developer partnership through the open code?
Boosting your offerings through partnering can increase your revenues.

Can you benefit from spreading knowledge and developing competence about your technology through Open Source?

- Can you benefit from easier hiring?
- A larger ecosystem?
- Can you acquire start-ups that are building on your technology?

Are there any risks to consider?

Will you expose any security flaws in your code that can be exploited?

Will competitors be able to copy important solutions? And if so, does it matter?

Other companies may not profit from your technology even if they can copy it, due to differences in business models, internal technology and competence or patent protection.

Can the project develop in directions that are unfavorable for you?

Are there any barriers to starting the Open Source project?

Do you own the rights to the code you want to release?

Are there any special security issues associated with releasing the code?

Preparations

Do you know about Open Source licenses?

Do you know how to relate to your community?

Do you know about governance in Open Source projects?

Do you know how to integrate Open Source code?

Do you want to transfer ownership of contributed code?

Are you prepared to change your internal development process to support Open Source development?

11 Bibliography

Abernathy, W. J. (1978). *The Productivity Dilemma.* Baltimore: Johns Hopkins University Press.

Andersson, C., & Runeson, P. (2007). A spiral process model for case studies on software quality monitoring - method and metrics. *Software Process: Improvement and Practice , 12* (2), 125-140.

Arthur, W. B. (1989). Competing Technologies, Increasing Returns, and Lock-In by Historical Events. *The Economic Journal , 99* (394), 116-131.

Arthur, W. B. (1996). Increasing Returns and the New World of Business. *Harvard Business Review , 74* (4), 100-110.

Audris, M., Fielding, R. T., & Herbsleb, J. D. (2002). Two Case Studies of Open Source Software Development: Apache and Mozilla. *ACM Transactions on Software Engineering and Methodology , 11* (3), 309-346.

Bach, J. (Octobre 1995, 2003). The Challenge of "Good Enough" Software. *American Programmer* .

Bergquist, M., & Lungberg, J. (2001). The Power of Gifts: Organizing Social Relationships in F/OSS Communities. *Information Systems Journal* (11), 305-320.

Black Duck. (11 May 2009). *Black Duck Suite*. Collected from Black Duck: http://www.blackducksoftware.com/black-duck-suite 11 May 2009

Capek, P. G., Frank, S. P., Gerdt, S., & Shields, D. (2005). A history of IBM's open-source involvement and strategy. *IBM Systems Journal , 44* (2), 249-257.

Carnegie Mellon Software Engineering Institute. (6 May 2009). *CMMI® for Development, Version 1.2.* Collected from Software Engineering Institute Carnegie Mellon: http://www.sei.cmu.edu/publications/documents/06.reports/06tr008.html 6 May 2009

Carnegie Mellon Software Engineering Institute. (6 May 2009). *Free Software Quality and Security Open Source Report.* Collected from Coverity: http://www.coverity.com/html/user_registration.php?doc=open_source_quality_report.pdf 6 May 2009

Chesbrough, H. W. (2003). The Era of Open Innovation. *MIT Sloan Management Review* , 35-42.

Choi, J. (1994). Network Externality, compatibility choice, and planned obsolescence. *The Journal of Industrial Economics , XLII* (2), 167-182.

Coff, R. W. (1999). When Competitive Advantage Doesn't Lead to Performance: The Resource-Based View and Stakeholder Bargaining Power. *Organization Science , 10* (2), 119-133.

Dahlander, L., & Magnusson, M. G. (2005). Relationships between open source software companies communities: Observations from Nordic firms. *Research Policy , 34* (4), 481-493.

Eclipse Foundation. (15 April 2009). *About the Eclipse Foundation.* Collected from Eclipse Foundation: http://www.eclipse.org/org 15 April 2009

Eclipse Foundation. (15 April 2009). *Eclipse Projects.* Collected from Eclipse Foundation: http://www.eclipse.org/projects 15 April 2009

Eclipse Foundation. (5 May 2009). *Eclipse Public License (EPL) Frequently Asked Questions*. Collected from Eclipse.org: http://www.eclipse.org/legal/eplfaq.php#BUSADVOS den 5 May 2009

Ehrensvärd, S. (20 February 2009). CEO. (J. Ekholm, & J. Johnsson, Interviewers)

Ehrensvärd, S. (5 May 2009). *Yubico.com*. Collected from Yubico.com: http://www.yubico.com/about/welcome/

Farrell, J., & Saloner, G. (1985). Standardization, compatibility, and innovation. *Rand Journal of Economics* , 70-83.

Goldman, R. (27 March 2009). Senior Staff Engineer and Open Source Adviser . (J. Ekholm, & J. Johnsson, Interviewers)

Google. (5 May 2009). *What is Android?* Collected from Android.com: http://www.android.com/about 5 May 2009

Höst, M., & Runeson, P. (September 2007). Checklists for Software Engineering Case Study Research. *Empirical Software Engineering and Measurement* , 479-481.

Hall, G., & Howel, S. (1985). The experience curve from the economist's perspective. *Strategic Management Journal* , 197-212.

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

Hecker, F. (1999). Setting Up Shop: The Business of Open-Source Software. *IEEE Software , 16* (1), 45-51.

IBM. (2007). *Annual Report 2007.* Armonk, NY: IBM.

IBM. (11 May 2009). *IBM Products*. Collected from IBM.com: http://www.ibm.com/products/us/en 11 May 2009

Ives, B., & Learmonth, G. P. (1984). The information system as a competitive weapon. *Communications of the ACM , 27* (12), 1193-1201.

Jullien, N. (2008). *Developing "FLOSS", a market driven investment.* Bretagne: LUSSI, Telecom.

Katz, M., & Shapiro, C. (1985). Network externalities, Competition and Compatibility. *The American Economic Review , 75* (3).

Katz, M., & Shapiro, C. (1992). Product introduction with network externalities. *The Journal of Industrial Economics* , 55-83.

Katz, M., & Shapiro, C. (1986). Technology adoption in the prescence of network externalities. *Journal of Political Economy , 94* (4), 822-841.

Krishnamurthy, S. (2003). *An Analysis of Open Source Business Models.* Washington: University of Washington, Bothell.

Krishnamurthy, S. (3 May 2002). Cave or Community? An Empirical Examination of 100 Mature Open Source Projects. *First Monday* .

Krogh, G. v., & Spaeth, S. (21 June 2007). The open source software phenomenom: Characteristics that promote research. *Journal of Strategic Information Systems* , 236-253.

Lakhani, K., & Wolf, R. G. (2003). Why Hackers Do What They Do: Understanding Motivation and Effort in Free/Open Source Software Projects. *MIT Sloan Working Paper* , 1 ff.

Lapre, M., Mukherjee, A., & Van Wassenhove, L. (1999). Behind the learningcurve: linking learning activities to waste reduction. *Management Sciences* .

Lepak, D. P., Smith, K. G., & Taylor, M. S. (2007). Value Creation and Value Capture: A Multilevel Perspective. *Academy of Management Review , 32* (1), 180-194.

Lerner, J., & Tirole, J. (2002). Some Simple Economics of Open Source. *The Journal of Industrial Economics , L* (2), 197.
Levy, F. (1965). Adaption in the production process. *Management Science* , B136-B154.
Müller-Prove, M. (19 February 2009). User Experience Architect. (J. Ekholm, & J. Johnsson, Interviewers)
Malone, T. W., & Laubacher, R. J. (September-October 1998). The Dawn of the E-Lance Economy. *Harvard Business Review* , 145-152.
Microsoft. (5 May 2009). *Meet Windows Mobile*. Collected from Windows Mobile: http://www.microsoft.com/windowsmobile/en-us/meet/default.mspx 5 May 2009
Nokia. (11 May 2009). *About Nokia*. Colected from Nokia.com: http://www.nokia.com/about-nokia 11 May 2009
Nokia. *Nokia in 2008.* Helsinki: Nokia.
Open Source Initiative. (12 February 2009). *Open Source Initiative*. Collected from Open Source Initiative: http://www.opensource.org/docs/osd 12 February 2009
Open Source Initiative. (12 February 2009). *Open Source Initiative*. Collected from Open Source Initiative: http://www.opensource.org/history 12 February 2009
OpenOffice.org. (22 April 2009). *About OpenOffice.org*. Collected from OpenOffice.org: http://about.openoffice.org/index.html 22 April 2009
Orski, M. (2007). *Öppen källkod i Sverige: beprövad teknik och mediahype.* Stockholm: Obok.
Osterwalder, A. (2004). *The Business Model Ontology - a Proposition in a Design Science Approach.* Lausanne: Universite de Lausanne.
Porter, M. A. (1985). *Competitive Advantage.* New York: The Free Press.
Raymond, E. S. (2001). *Katedralen och basaren: en oavsiktlig revolutionärs tankar kring Linux och öppen källkod.* (R. Sjölander, Övers.) Nora: Nya Doxa.
Robson, C. (2002). *Real World Research* (Vol. II). Oxford: Blackwell Publishers.
Rosén, T. (2008). *Open Source Business Model - Balancing Customers and Community.* Institute of Technology, Department of Management and Engineering. Linköping: Linköping University.
Rossi, M. A. (2004). *Decoding the "Free/Open Source (F/OSS) Software Puzzle" a survey of theoretical and empirical contributions.* Dipartimento di Economica Politica. Siena: Quaderni.
Sahal, D. (1981). *Patterns of Technological Innovation.* Reading, MA: Addison-Wesly.
Schilling, M. (1999). Winning the Standards Race: Building Installed Base and Availability of Complementary Goods. *European Management Journal , 17* (3), 265-274.
Smith, D. (3 March 2009). Director of Eco System Development. (J. Ekholm, & J. Johnsson, Interviewers)
Spiller, D., & Wichmann, T. (2002). *FLOSS Final Report - Part 3.* Berlin: Berlecon Research.
Stake, R. (1995). *The art of case study research.* Sake.
Sun Microsystems. (16 April 2009). *Company Profile*. Collected from Sun Microsystems: http://www.sun.com/aboutsun/company/index.jsp 16 April 2009
Symbian Foundation. (11 May 2009). *About the Symbian Foundation*. Collected from Symbian Foundation: http://www.symbian.org/about.php 11 May 2009

Thum, M. (1994). Network externalities, technological progress, and the competition of market contracts. *International Journal of Industrial Organization* , 269-289.
West, J. (June 2003). How Open is Open Enough? Melding Proprietary and and Open Source Platform Strategies. *Research Policy* , 1259-1285.
West, J. (2007). Value Capture and Value Networks in Open Source Vendor Strategies. *40th Annual Hawaii International Conference on System Sciences.* San José: IEEE Xplore.
von Hippel, E. (2005). *Democratizing Innovation.* Cambridge: MIT Press.
Wood, D. (6 March 2009). Catalyst and Futurist. (J. Ekholm, & J. Johnsson, Interviewers)
Yelle, L. (1979). The learning curve: historical review, and comprehensive survey. *Decission Sciences* , 302-328.
Yin, R. (1994). *Case Study Research - Design and Methods* (Vol. II). SAGE Publications.

Appendix: Template for Interview Questions

What is your name and what is your position within the company?
Tell us about the company and what you do.
What is your core competence?
How is your core competence translated into a product offering?
What is (the specific project)?
Why did you go Open Source?
What have you gained/expect to gain from your Open Source involvement?
What are the downsides by going Open Source?
What risks do you see?
How has the competition changed?
What cannels do you use within the project?
What kind of relationship do you have with the stakeholders, such as customers and developers?
Who is developing the code?
Who are your customers and how has this picture changes since you went Open Source?
Are you partnering with someone?
What do you get revenue from?
What are your costs?