Open Source Vendors’ Business Models

Thomas Rosenfall

April 2012

Department of Management and Engineering
Linköpings universitet, SE-581 83 Linköping, Sweden
© Thomas Rosenfall, 2012

"Open Source Vendor’s Business Models"

Linköping Studies in Science and Technology, Dissertations, No. 1438

ISSN: 0345-7524

Printed by: LiU-Tryck, Linköping

Distributed by:
Linköping University
Department of Management and Engineering
SE-581 83 Linköping, Sweden
Tel: +46 13 281000
Acknowledgements

This thesis was a long time in coming. The author is graciously indebted to several individuals and organizations for their support and contributions during the process.

First, the work of my supervisor, Anna Öhrwall Rönnbäck, together with professor Staffan Brege, whom probably are the sole reason for the completion of this thesis. Without their support and advice, the author would have given up years ago.

The author would also like to thank the case companies and their representatives that have shown patience over the years, with a special note to the new and former management of Cendio AB.

Several other colleagues have been involved in the process and provided much support and insight, for which the author is grateful. A special “thank you” goes to Christina Grundström for the support and advice concerning as assistance supervisor. Additionally, the author graciously would like to thank Mica Comstock for the outstanding proofreading.

The research was funded by Lundbergstiftelserna in 2005 - 2008, while the final thesis was conducted as a part of the research project BOSSANOVA, financed by VINNOVA between 2008 and 2011.

Finally, the author would like to thank friends and family, in particular Åsa, for support and encouragement that was sorely needed.

Please note that the author changed his surname to Rosenfall from Rosén on August 1, 2011.

Linköping, 2012
Thomas Rosenfall
8.1. Business position ......................................................................................................... 99
8.2. Operational platform .................................................................................................. 107
8.3. Offering ........................................................................................................................ 109
9. Case 4: Red Hat Inc. ..................................................................................................... 111
  9.1. Business position ....................................................................................................... 111
  9.2. Operational platform ............................................................................................... 118
  9.3. Offering ..................................................................................................................... 120
10. Analysis ..................................................................................................................... 123
  10.1. Business position analysis ..................................................................................... 123
  10.2. Operational platform analysis .............................................................................. 124
  10.3. Offering analysis .................................................................................................... 125
  10.4. Cross-case analysis ............................................................................................... 127
  10.5. Generic configurations .......................................................................................... 136
11. Conclusions ............................................................................................................... 141
  11.1. Business model framework ................................................................................... 141
  11.2. Business models ..................................................................................................... 143
  11.3. Generic business model configurations ............................................................... 145
  11.4. Profitable and sustainable configurations ............................................................ 148
12. Discussion and future research .................................................................................. 151
  12.1. Future research ...................................................................................................... 153
References ...................................................................................................................... 157
Appendix 1: Glossary
Appendix 2: List of interviews
Appendix 3: Interview guide 1 (Swedish)
Appendix 4: Interview guide 2 (English)
Appendix 5: Interview guide 3 (English)
Appendix 6: Phone survey
1. Introduction

Open-source software (OSS) is software that is developed by voluntary development contributions, and is freely available for everyone to download, use, re-distribute and change. As a term, OSS has existed since 1998 (Raymond, 1998), but as software since 1984 (Raymond, 2001; Stallman, 2002). The developers are organized in projects and communities around specific software. Despite the differences between them, all projects and communities are collectively called the OSS community.

Under these circumstances, companies which offer software products based on OSS face significant challenges, since the software is available for everyone, including competitors, which makes traditional protection for software, such as intellectual property (IP), patents or closed source useless. In addition, many of the OSS licenses remove the possibility for direct revenues from the software, i.e. licenses from traditional software companies. Moreover, many OSS projects are averse against commercialization of their software. In this thesis, firms that offer software products based on OSS are henceforth called “OSS vendors”. What OSS vendors offer are essentially software packages, which are based on OSS, together with services such as updates and support. In order to be labeled as an OSS vendor in this thesis, these products must be the main income to differentiate them from other OSS-related businesses. While other companies may offer OSS-related products, such as consultancy services and hardware bundling, their main income is not derived from offering OSS-based software products; consequently, such companies are not considered OSS vendors.

OSS vendors try to balance between community interaction and customer demands, whose goals might not match or be outright disparate. The stakes are high; to be successful, OSS vendors have to offer attractive commercial products without alienating the OSS community (Rosén, 2008).

Over the years, OSS has gone from being the hobbyist’s alternative to mainstream and commercial software (Clarke, 2010a). Different aspects of OSS have been studied in academic research, such as the impact on innovation (e.g. von Hippel and von Krogh, 2003; cf. Klineciewicz, 2005), or motives for OSS developers (e.g. Lerner and Tirole, 2002; Ghosh, Krieger, Groot and Robles 2002a; Bonaccorsi and Rossi, 2003a). While these studies concern those who contribute to OSS, others concern the economics for organizations using OSS (Lerner and Tirole, 2002; Mustonen, 2003), and business models for OSS companies (Lerner and Tirole, 2002; Wichmann and Stiller, 2002; Dahlander, 2005; Krishnamurthi, 2005), including how OSS vendors license their products (Välimäki, 2003; Comino and Manenti, 2007).

Although previous research has examined many different aspects of OSS, there is still a research gap concerning how OSS vendors conduct their business. To a large extent, the early research concerning business models for OSS was both presented and influenced by non-scholars well versed in the OSS community (e.g. Hecker, 1999; Raymond, 2001, Goldman and Gabriel, 2005; Perens, 2005). These observers were naturally more concerned about practical implications than theoretical reasoning.

The early OSS business models presented by Raymond (2001) were somewhat limited in scope, focusing on offerings in general and revenue models in particular. The connection between the firm and its offering and the community is seldom, if ever, discussed. In addition, some of the proposed business models suffer from the lack of empirical testing. Yet, these
business models have later influenced researchers to further theorize about OSS business models in the same way (e.g. Wichmann and Stiller, 2002; Lerner and Tirole, 2002; West, 2003; Krishnamurthy, 2005).

To understand how OSS vendors conduct their business, it is essential to look beyond the offering and revenue models into a more holistic concept, adding the community into the mix. It is also important to empirically study companies to determine their business models. Finally, the potential tension between idealistic communities and performance-oriented customers could hypothetically impair the profitability for OSS vendors. Why have some managed to be profitable, while others have failed?

1.1. Purpose

The purpose of this thesis is to contribute to the understanding of how OSS vendors conduct their business. More specifically, this thesis aims to:

- Describe how OSS vendors’ business model can be configured;
- Investigate if there are generic patterns in OSS vendors’ business models configurations, and if so, describe and analyze these; and
- Investigate if there are configurations that are more profitable and sustainable than others, and if so, describe and analyze them.

1.2. Outline of the Thesis

The thesis is outlined as follows. After this section, the methodology and research design is presented, followed by two theoretical backgrounds, one on business models and the other on open-source software.

Following the theoretical backgrounds, the business model framework used in the research is presented. This section is followed by the four cases, which are presented in alphabetical order. The cases both consist of gathered information, together with some in-case analysis.

The cases are followed by analyses, conclusions, discussion and recommendation for further research.
2. Methodology

In general, the research is empirically based with the aim to discover emerging patterns, as a “normal cycle of research” (Meredith, 1989). However, conceptual ideas and perspectives in the theoretical outline were used when constructing questions for the interview guides and for the survey (Miles and Huberman, 1994).

Qualitative research is recommended when exploring phenomena about which little is known or to gain novel understanding of existing phenomena (Stern, 1980), such as intimate details of a specific phenomenon (Strauss and Corbin, 1998). Since prior research on OSS vendors was relatively scarce, the field needed to be explored, why a qualitative methodology was used to generate theory (Eisenhardt, 1989). The number of cases was selected from a relatively small population, and the sample was deliberately disparate to cover different aspects of the phenomenon, but also to try to generate generic theory (Yin, 1994; Eisenhardt 1989; Eisenhardt and Graebner 2007).

2.1. Research design

The research design for this thesis has therefore been based on iterations. The study was inspired and thus initiated as a result of the author’s knowledge about this subject, followed by theoretical retrieval, before the first set of interviews and other empirical data collection were conducted. The purpose was to find a research gap and find relevant research questions (Eisenhardt and Graebner, 2007). Interviews and empirical data collection were conducted in three stages, with iterations between data collecting and returning to existing literature.

The first stage of interviews used an already-tested interview guide, derived from an earlier theoretical framework. The reason for this was that an opportunity occurred, where key executives became available for interviews. Therefore, the decision was made to make use of an existing framework to gather initial information. However, when the result of the interviews were reviewed, it became clear that this guide did not cover essential matters, for example the role of the OSS community, when addressing the OSS industry. Therefore, based on the findings and complementary literary studies, a new framework was constructed with a new interview guide. Conducting research in this manner enables researchers to focus on aspects of choice and interest rather than becoming overwhelmed by the empirical data from the cases (Yin, 2003). The same iteration was made between the second and third stage of interviews.

2.1.1. Logical reasoning

The research has been conducted in stages, with literature searches between each stage, and with knowledge added in each stage. The logic reasoning has been described as abduction (Alvesson and Sköldberg, 1994). Since the author worked in the OSS industry, he entered the research process with prior knowledge, as described above. In addition, together with the preknowledge also followed preconceptions about various aspects of the said industry. During the research process, some of these preconceptions have been questioned and dismissed, while others have been reinforced. Thus, it is fair to say, that aside from the abductive process, some induction, based on pre-knowledge, facts and pre-conception have occurred. However, since the research is more exploratory, creating hypotheses has been important to gain forward momentum. As part of the abductive concept, answers may not be all-inclusive, but are bound to change or be revised with new facts. The author hopes that this thesis will answer to how OSS vendors conduct their business, but also awareness that new questions
will arise. The OSS industry is also part of the fast-changing IT industry, which might alter the market conditions on short notice and thus generate new questions.

### 2.1.2. Case selection

After the first interview stage, some consideration was made to streamline the case companies with similar business concepts, namely OSS vendors. OSS vendors are companies that offer software products, which are based on OSS as their primary source of revenues. The products usually consist of OSS bundled with services, such as selected and packaged software, installation and configuration scripts, certifications, testing and support. The resulting products are usually labeled under specific brand names.

The empirical data consists of interviews from selected OSS vendors, specifically Cendio AB, CodeWeavers Inc., MySQL AB (later part of Sun Microsystems Inc., and Oracle Inc.), and Red Hat Inc. The companies were selected based on five criterions: a) they were OSS vendors; b) the companies, aside from the products, developed or maintained OSS; c) the companies were based in either Sweden or the United States; d) they were well-established and relatively mature (in existence for at least five years); and e) they provided relatively easy access to interviews and material.

Data collection and data analysis did, to a certain extent, overlap and thus resulted in a fruitful interaction (as argued by Eisenhardt, 1989). Based on in-depth interviews, rich case studies (see Gummesson, 1991) have been developed. In this study, such cases are presented as stories. These stories cannot convey all the information captured in the interviews, but give the reader an overview of the companies.

### 2.1.3. Limitations

One important methodology aspect was the number of interviews with each case company. The size of the companies limited the number of interviewees who were able to answer the questions. Another limiting aspect was access to knowledgeable executives in the larger companies.

To balance the information collected from the small versus medium and large companies, the decision was made to conduct fewer, but longer and more detailed interviews, with a limited number of key respondents (highly-placed and knowledgeable about the history, current strategy and operations of the firm) at each company. How the data was collected in the four case companies will be detailed in the data collection section presented below.

### 2.2. Data collection

The data was collected in three stages. The first stage was a pre-study conducted during 2005 which started with an impromptu interview with CodeWeavers Inc. in April, travel to the United States in November and interviews in Sweden in December.

#### 2.2.1. Quality of data

All of the interviews in the first two stages were recorded using digital recorders. Most of that material was transcribed in order to make it easier to access and use. All of the recorded material was archived for later use. The data from the third-stage interviews was not digitally recorded, but the answers were written down. Since some of the firms in the interviews asked to be anonymous, all the raw data was archived, but access is restricted. The compiled data was available upon request. In addition to the interviews, access was granted by three of the four case companies for internal documents, as well as official material from the Internet and
other sources. Red Hat Inc. was unable to comply with this request due to regulations. All data has been stored in its raw form together with the refined material. According to Yin (1994), the successful administration of all the data collected is one of the main advantages for using case studies.

The main reason for using case studies was that the research was in an exploratory stage, with a need to explore the industry. By using a business model framework, insights were gained into the inner workings of the firms in the OSS industry. These insights then enabled us to further explore and interpret some of the results acquired (Merriam, 1994), as described further in the analysis and conclusion chapters.

Another important matter was the number of possible companies to study. The majority of the Swedish OSS firms consisted of one person, usually working as a consultant, and solely working with OSS. One of the concepts was to study differences, if any, between Swedish and American companies. A major difference was the size of the companies and their potential customers. As the FLOSS study clearly showed, the use of OSS in Sweden, especially among governmental institutions, was (in 2002) one of the lowest in Europe (Wichmann, 2002a), thus providing less business opportunities for OSS vendors. Therefore, since the main buyers of OSS products and services traditionally have been governmental institutions, the lack of such customers in Sweden could be one the reason for the low number of national OSS vendors.

Consequently, this research may be seen as relatively lacking in empirical material, but the lack of prior research, combined with one of the purposes was to create a theoretical framework to study these kinds of firms, the case materials should be sufficient. In addition, the firms have been studied for several years, providing a depth of data to study.

It is the view of the author and others (Pettigrew, 1997) that most events and research are a process. Therefore, all the interviews and associated results have been communicated to the companies. In some cases, this has generated internal discussions regarding customers or pricing issues. Moreover, to ensure the quality of the material, project members, academic supervisors and colleagues reviewed the case material and creating the framework model and interview guides to ensure quality.

### 2.2.2. Primary data and interviews

The research data consisted of interviews structured with the help of an interview guide, often referred to as a “semi-structured approach” (Yin, 1994). The respondents, typically executives in managerial positions within the case companies, worked with the business aspects of developing products. The defining aspect for case selection for this study was that the offering had to be based on OSS. The questions were open-ended and allowed the respondents to formulate their answers using their own words.

Most interviews were conducted with two researchers, but in some cases a single researcher conducted the second-stage interviews. However, each interview was digitally recorded in order to reduce the risk of misinterpretations in the subsequent case write-up and analysis. Additionally, both introductory and complementary e-mail conversations were conducted between the researchers and the respondents concerning the study. To further minimize this potential problem, the cases were fed back to key respondents and their changes (in some cases only minor) were addressed and included in the case description.

### 2.2.3. Secondary data

Some secondary data was collected from peer-reviewed journals, articles or published books. However, the majority of the secondary data was primarily collected from the Internet, consisting of financial reports, information from company websites, and commentaries (blogs and
chronicles) or articles from online publications. The OSS phenomenon is an online move-
ment, which is the reason for the prominence of online sources.
However, the validity of blogs may be debated; therefore, these sources were largely used
to illustrate or provide another viewpoint for a company or project. Nevertheless, the sources
used have been selected with some care, where the most important contributors are well con-
nected and versed in OSS, which provides an outside perspective (Eisenhardt and Graebner,
2007). In this thesis, many comments are from prominent bloggers or writers. Two of the
most cited writers are Matt Asay and Steven Shankland. Matt Asay has worked with OSS
since 1998 as an executive in companies like Novell, Alfresco and Canonical. He has blogged
about OSS for many years, and started to write for The Register in 2010 (Asay, 2010a). Ste-
phen Shankland is a senior writer for CNET and has written about IT since 1998 (Shankland,
2011).

2.2.4. The first-stage interviews

The first stage of interviews was conducted in 2005 and 2006, where four Swedish companies
and four American companies were interviewed in their respective countries. The purpose
was to explore how business based on OSS was conducted in Sweden and the United States.
The Swedish companies were Cendio AB, MySQL AB, Redbridge AB, and RedPill-LinPro
AB. The first two offered software products based on OSS, while the last two were consultan-
cy firms specializing in OSS.
The American companies were CodeWeavers Inc., IBM Software Inc., Red Hat Inc., and
Sun Microsystems Inc. These companies all offered software products, however IBM Software
Inc. and Sun Microsystems Inc., as multinational corporations, offered a wider range of
products and services, and their OSS-related revenues were minuscule compared to their prop-
rietary and hardware business.
With the exception of CodeWeavers Inc., the American companies were generally much
larger than their Swedish counterparts. The largest Swedish company in the study, MySQL
AB, had fewer employees worldwide than IBM and Sun employed in Sweden.
The outcome and findings of the pre-study, therefore, were inconclusive. This was partly
because the theoretical framework utilized was not properly designed for important aspects of
OSS. Moreover, the Swedish phase of interviews was cut short when the deficiencies could
no longer be ignored. A new business model framework was needed to fully understand how
the firms operated and to create a more suitable interview guide.
The other part that deemed the study inconclusive was the significant differences between
the business logic among the interviewed companies.
The large multi-national companies like IBM and Sun Microsystems used OSS as part of
their marketing for other business, subsidizing their OSS efforts with the revenues from their
non-OSS divisions. Both IBM and Sun Microsystems saw added benefit from OSS based on
the perceived fast-paced innovation from the OSS community.
IBM and Sun Microsystems differed on how OSS was utilized. IBM offered certified Linux
distribution for their entire server line complete with released drivers for all hardware compo-
nents. In addition, IBM is active in several OSS projects such as Apache and Eclipse, spon-
soring with development, hardware, patents, and monetary contributions to developer events
and fairs. Both Apache and Eclipse have permissive licenses, enabling IBM to offer propri-
etary versions. IBM does not own the copyright for any of the software. IBM founded the E-
lipse project together with other software vendors and donated the software and intellectual
property (IPR). A neutral not-for-profit foundation was founded in 2004 to govern the project
in order to make it vendor neutral. IBM has no legal or formal ties to Apache aside from
sponsoring the project through developer time.
On the other hand, Sun Microsystems owned the copyrights for most of their OSS, such as Java, OpenOffice, and OpenSolaris. Sun also offered proprietary versions of their OSS software, such as Java Mobile Edition, Java Standard Edition, Java Enterprise Edition, StarOffice, and Solaris. The differences between the open and proprietary versions were added support services, bundled proprietary third-party software, and OEM possibilities. However, Sun Microsystems struggled between the demands of the OSS community in terms of licensing and the extra value customers received from the proprietary products.

The consultancy firms were enthusiastic about OSS, but seldom contributed to any project. Their main business was to offer competence to customers, which tended to be more hands-on, rather than development-oriented. Their solutions to customer problems mainly consisted of programming, but often in terms of short scripts that solved the immediate problem. However, most of the consultancy companies allowed their employees to be active in OSS projects on non-chargeable time. Moreover, most of the consultants were likely to work with proprietary software instead of OSS.

2.2.5. Second-stage interviews

After reviewing the results from the first stage, it was decided that the OSS firms offering software products were the most interesting to study, since they not only worked with communities but also were also dependent on them. The four software companies already interviewed, Cendio, CodeWeavers, MySQL, and Red Hat. The sample was based on two criteria. First, that Red Hat was the undisputed OSS market leader and the largest OSS company in United States (and the world), and that MySQL held the market leader position for OSS databases and was the largest OSS Company in Sweden. Second, due to the author’s unique insights into Cendio, it was a given that this sample be included. In addition, the selection of one large and one small company from each country could facilitate comparisons (Eisenhardt and Graebner, 2007), which was the reason for adding CodeWeavers. For these reasons, the selection of cases should be considered a good sample (Yin, 1994).

The data collected in the first and second stage of interviews was used in the author’s licentiate thesis (Rosén, 2008).

2.2.6. Third-stage interviews

The third stage, conducted in fall 2009, was different. In this stage of interviews, member companies of the Open Source Sweden association were approached and data was collected in a standardized phone interview. The questionnaire (Appendix 6) was based on the findings from the licentiate thesis, and the purpose of the interviews was to validate the findings.

The study was initially constructed as a survey, but responses from early interviewees indicated that the questions needed extra explanation provided by the interviewer. The interviews were conducted by phone, using a survey-like questionnaire. Out of 32 members, 31 were contacted and 27 full and one (1) incomplete interview were collected. Each interview lasted about 30 minutes. Even if the majority of the questions were constructed as a normal survey question with a range of seven choices, other questions were open-ended.

Three of the firms from step two also were members of the association; Cendio, MySQL, and Red Hat representatives were also interviewed in this step. Since CodeWeavers were not interviewed, extra information was collected by e-mail. Since most of the interviews in step two were conducted in 2007, the new information was used to update the cases.

3 The incomplete interview was not conducted by phone; instead, the respondent sent the questionnaire via e-mail. This meant that many questions were misunderstood and other answers were incomplete.
As stated before, in addition to the structural or semi-structural interviews and the secondary data collected from various sources, the author has been in continuous contact with Cendio since his departure from the company in 2005. These contacts have been consultative, as Cendio has been interested in learning how to change its business. Due to the delicate nature of the information, these meetings and e-mails have been kept informal, as requested by the company. The implications of these contacts have enriched the knowledge about Cendio’s business processes, but there is also the risk of bias. However, the scope of the thesis should minimize the effects of such partiality.

2.2.7. Data collection per case company

2.2.7.1. Cendio AB

As already stated, the author was employed in several executive positions at Cendio AB between 1999 and 2005. After the employment ended, the author and the management of Cendio AB have kept in contact through informal meetings and mail conversations.

Formal information was gathered in 2005 through two interviews, one with the CEO of that time and one with the founder. The interviews were conducted by a neutral researcher with the author as a passive listener, using the interview guide in Appendix 3. These were digitally recorded and transcribed. In addition to the two interviews, secondary data was collected from written material provided by the company, as well as from various electronic sources. In addition to the formal and secondary data collection, unofficial information was gathered through several informal meetings with the management. All data was consolidated into a case description, which was sent to the CEO and founder for correction. The concluded case description was used as part of the author’s licentiate thesis (Rosén, 2008). The thesis was sent to the participants, and representatives from Cendio were present during the presentation of the thesis in May 2008.

In 2009, the former CFO became CEO and the former management team was dissolved. The informal contacts increased as a result of the changes. Since Cendio AB is a member of Open Source Sweden, the CEO partook in the round of interviews. However, instead of conducting the interview by phone, the interview, Appendix 6, was conducted in person and notes were collected. Due to technical problems, the digital recording failed, so the written records are the only available material from the interview.

In addition to the formal and informal data, secondary data was collected from electronic sources. Cendio AB has also been studied by other researchers (Dahlander and Magnusson, 2005; 2008), and some of their data has been used to validate some of the findings.

All collected data, including the data collected for the licentiate thesis, was compiled into a case description. This material was sent to the CEO for amendment. The finalized case description used in the thesis was approved and validated by the CEO.

2.2.7.2. CodeWeavers Inc.

The COO of CodeWeavers Inc., the first interviewed case company, visited Sweden in April 2005. This interview was conducted just after an interview with a Swedish computer publication (Åsblom, 2005). The decision to interview an American OSS company set the direction for the first stage interviews. The impromptu and hurried interview was conducted by three researchers and made use of an existing interview guide, directly translated from Appendix 3. It was digitally recorded and transcribed.

Between 2005 and 2007, there were sparse mail contacts between the firm and this author. In December 2007, the founder and CEO and the COO were formally interviewed at the company headquarters in St. Paul, Minnesota, using the interview guide in Appendix 5. This interview was conducted by the author, and was digitally recorded and transcribed. In addition
to the formal interview, secondary information was collected from brochures, but mainly electronic sources. The collected data was assembled and fed back to the company for confirmation. With some minor changes, the information was compiled into a case description, which was a part of the licentiate thesis (Rosén, 2008).

Between 2007 and 2009, sparse mail contact was maintained between the author and the company; for example, the author tested and evaluated CodeWeavers’ product for Mac. In the beginning of 2010, data was collected via mail to be used in this thesis. The data was compiled into a case description and sent to the COO for modification. The corrected version is used in this thesis together with all the other data from earlier data collections.

2.2.7.3. MySQL AB

MySQL AB has, aside from Cendio AB, provided the richest data, mainly from interviews. In the first stage, in 2005, the CEO was interviewed at the San Francisco office, using the interview guide from Appendix 3. The interview was digitally recorded and transcribed.

In 2006 and 2007, one of the co-founders and the manager of community relations were interviewed twice, in Stockholm and London, using a Swedish translation of the interview guide in Appendices 4 and 5. The interviews were conducted by two researchers, and digitally recorded but only partly transcribed, due to sound problems. In addition to the three interviews, secondary data was collected via electronic sources. The data was assembled and fed back to the company, and used in the licentiate thesis (Rosén, 2008).

When MySQL AB was acquired by Sun Microsystems Inc. in 2008, both of the interviewees left the company. Therefore, during the third stage of interviews, a new informant was interviewed by phone from Germany, using the interview guide in Appendix 6. This interview was not digitally recorded, but written down. In addition to the phone interview, sparse email conversations followed. Secondary data was also collected through the company website and other electronic sources.

At the same time, Sun Microsystems Inc. was under acquisition by Oracle Inc., and when the deal was completed in 2010, the new interviewee also left the company. However, the interviewee agreed to validate the data in 2010, before it was used in this thesis. All data from all the three stages of interviews and data collection have been compiled and is used in this thesis. Moreover, since MySQL AB and Cendio AB were engaged in the same research (Dahlander and Magnusson, 2005; 2008), some of that research was used to validate the data collected in this thesis.

2.2.7.4. Red Hat Inc.

Red Hat, being a publicly-held company, is governed by the rules of the New York Stock Exchange (NYSE). Therefore, in the information gathered from the company, aside from the formal interviews, no informal data was collected.

In the first stage of interviews in 2005, the CTO was interviewed at the company headquarters in Raleigh, North Carolina. The interviewee, aside from his position in the company, is one of the most influential individuals of OSS, since he was also President of the Open Source Initiative (OSI), and also co-founded Cygnus Solutions in 19894. The interview, which followed the interview guide in Appendix 4, was conducted by two researchers and digitally recorded and transcribed. The compilation was also sent to the informant for validation.

---

A second interview was conducted with the same interviewee in 2007, also at the company headquarters, this time using the interview guide in Appendix 5. This interview was conducted by the author, and once again was digitally recorded and transcribed. In addition, secondary data was collected from the company website, together with other electronic sources. The data was compiled as a case description and sent to the informant for validation. The result was then used as part of the licentiate thesis (Rosén, 2008).

Since Red Hat is a member of Open Source Sweden, the Swedish CEO became the informant in the third stage of interviews, and was specifically asked to provide a Swedish perspective. The phone interview followed Appendix 6, and the answers, while not digitally recorded, were written down. In addition to the phone interview, secondary data was collected to update the corporate information. All data was compiled and sent to the Swedish CEO for validation. All data collected, including the data used in the licentiate thesis, have been used in this thesis.

2.3. Quality of the research

Scholars have suggested construct validity, internal (or logical) validity, external validity, and reliability as a way to describe how rigid a research study is (Yin, 1994; Gibbert, Ruigrok and Wicki, 2008).

2.3.1. Internal validity

Yin (1994) states that internal validity concerns the causal relationship between events and the result. The risk is that factors or events, known or unknown to the researcher, “contaminate” the result. Yin suggests that the researcher aims to construct explanations and try to find patterns in the data (1994). In this thesis, a research framework is used to illustrate how the analysis was done and the outcome of it (Gibbert, et al., 2008). Moreover, the cross-case analysis aims to construct explanations based on patterns in the collected data (Yin, 1994, Eisenhardt, 1989). Additionally, there have been iterations between the findings, in the different interview stages, and theoretical literature, to conduct theory triangulation (Yin, 1994, Gibbert, et al., 2008).

2.3.2. Construct validity

Construct validity concerns how well the theory and model matches the study object (Yin, 1994). The early stages of the study suffered from the lack of a proper model and the use of incomplete theories. The iteration between data collection and theory was deemed necessary to achieve higher construct validity, in order to explain the chain of evidence (Yin, 1994), but also to compare between the findings and theories (Eisenhardt and Graebner, 2007). Another way to gain higher construct validity is to utilize “triangulation”, where multiple sources are used to confirm a phenomenon (Yin, 1994). In this thesis, triangulation was exercised by interviewing different executives in the case companies several times, and by making use of independent industry observers’ coverage of events in the industry and the case companies, where applicable. In addition, since data has been collected in different stages and at different times together with applicable secondary data, internal validity is also verified (Gibbert, et al., 2008).

2.3.3. External validity

External validity concerns the ability to generalize beyond the case study (Yin, 1994). This can be achieved by replication, which is proven by replicating the result through other studies.
However, this study is exploratory, so external validity is achieved by a selective sampling of cases, with four cases consisting of both market leaders and smaller firms, and unique insights (Yin, 1994), combined with a comparison between the said market leaders and smaller companies (Eisenhardt and Graebner, 2007). In addition, a cross-case analysis should provide basis for some generalizations (Eisenhardt, 1989). Moreover, the analytical framework, with clear examples, should enable replication of the study. The ability to conduct a second study, based on the research framework, also concerns the reliability of the study.

2.3.4. Reliability

Reliability concerns the ability of another researcher to repeat the study, and reach the same conclusions or results from the data (Yin, 1994). In this thesis, the business model framework was constructed with repeatability in mind. The data collected in three of the cases, Code-Weavers Inc., MySQL AB and Red Hat Inc. should be accessible for another researcher as suggested by Yin (1994). However, the relationship with Cendio AB enabled the author to gain a deeper understanding about that company, which could be hard for another researcher to obtain. This is discussed on the next section.

In addition, the names of the case companies are mentioned, which enhances the possibility for other researchers to study and replicate the study (Gibbert, et al., 2007).

2.3.4.1. Reliability risks

As a researcher, there is a risk of knowing “too much” about the study object, which can result in bias. During the course of the research, professional contact between the author and Cendio’s management team was intact, and internal business matters were discussed. This gave the author deep insight into Cendio’s operations, which would be difficult, if not impossible, for an outside researcher to achieve. However, this enabled him to find interesting research topics (Eisenhardt, 1989). However, the author has been aware of the risk of bias, and findings have been sought to be validated by colleagues and third-party sources. For example, another researcher conducted the interviews at Cendio AB with the author as a passive listener.

There is also a risk that the researcher misunderstands or interprets an answer incorrectly. Since this research was conducted in both Swedish and English, and because that the author’s native language is Swedish, misunderstandings could have occurred when interviews were conducted in English. Additionally, since the author is hearing impaired, the risk for misunderstanding was greater.

In order to address these risks, the majority of the interviews were digitally recorded, transcribed or summarized in writing. However, two recordings were lost due to technical problems, thus excluding them from this study. Moreover, all the case companies presented in this research had the data collection or case descriptions sent to them for review. The material used for the analysis has been approved by these companies.

2.4. The role of the researcher

It could be argued that the researcher is central in qualitative research, which is the method in this thesis. This is because the researcher is responsible for all of the work in terms of collecting, selecting, analyzing and interpreting data and in formalizing the collected information (e.g. Merriam, 1994).

In general, the philosophy of science states that there are two competing positions in research, positivistic and hermeneutic. Some claim that a researcher can only hold one of these positions (e.g. Andersson, 1979). In some research traditions, the concept of the researcher is
that of a “blank slate” or *tabula rasa*, onto which knowledge comes through unbiased observation (Strauss and Corbin, 1998).

However, this would also imply that a researcher has a limited view on the subject (or object) of research and uses the methods and tools that are attributed to the particular position. In empirically-based research, there is a relationship between different systems, reality, the researchers and actors (Normann, 1975), rendering objective and unbiased “facts” impossible.

Moreover, it could be argued that a researcher may change research traditions, which could lead to a mixed scientific view. Other scholars share this perception; for example, Gummeson (1991) argues that a researcher can adopt different research philosophies. Moreover, other aspects, such as driving forces, interests, education, and experiences affect most human beings, including researchers.

### 2.4.1. Systems theory positioning

A fair summary of the author’s scientific view, based on the different driving forces, is one of the systems theory combined with an analytical perspective (*c.f.* Arbnor and Bjerke, 1994). The positivistic concept of everything being measurable and labeled, combined with the ability to find recognizable patterns, is central to this view. Where other disciplines, such as mathematics and physics, manage to reduce events into formulas and theories, the discipline of engineering and management strives to reduce the field of companies into formulas and all-encompassing theories (for example gaming theory and the plethora of “theory of the firm”). However, there is one aspect that makes this effort nearly impossible: the fact that firms and business are operated by a chaos factor, i.e. humans.

Even so, a hermeneutic interpretation of a single entity may not further the understanding of a research object (Eisenhardt, 1989). Research should strive to enlighten, explore and further the understanding of the studied object, but also seek to extend this understanding to use, so that practitioners can benefit from the research. Fel! Hittar inte referenskälla. below illustrates the author’s scientific view on the research using the three strategies of truth, where: *correspondence* refers to external events for truth; *meaning* to internal deeper meaning for truth; and *application* to the aspiration of making use of the truth (Alvesson and Sköldberg, 1994).

![Figure 1: The scientific view of the author, based on Alvesson and Sköldberg (1994, p. 36).](image)

### 2.4.2. Driving forces and interests

It can be argued that there have to be some sparks of interest or curiosity for one to become a researcher. This is true for the author, where curiosity, paired with a need to know as much as
possible, are two of his primary driving forces. These two forces have been the reason for most of his life decisions, in particular, education (see below).

Computer science is another important interest of the author, having been introduced to him in the seventh grade in 1980. The driving force behind his computer interest was gaming; the author has had personal computers since 1986, all purchased with gaming in mind. In the beginning, games were made available through purchased tapes or cartridges. However, the market for games was new at that time, offering few and expensive products. Most of the games were distributed from user to user, by tape or printed source code. Some dedicated magazines also distributed games through source code. Because of this, the author became interested in programming, beginning with BASIC\(^5\), and later assembler language\(^6\). At the university, his interest turned to online gaming, through one of the computer societies, Lysator\(^7\) and the text-based NannyMUD, where the game was developed in a programming language called LPC.\(^8\)

Another of the authors' interests is reading books, and this became one way for him to explore new worlds and expand his mind. His favorite themes are science fiction and fantasy, and the author spent many hours in the library, quickly exhausting the limited number of available books. The lack of translated novels to Swedish forced a transition to English literature, which furthered the number of works and extended his English vocabulary. The author retains a library consisting of more than 1500 books, mostly consisting of science fiction and fantasy in English. This interest affected the author's selection of university-level courses,

2.4.3. Education

The author’s interest in gaming influenced his choice of high school, as the science high school he chose offered computer science as an option. Even if the overall scores were below average, the scores for computer science and other non-science courses, like religion and history, were much higher.

The author's university studies were based on his reading interests, with degrees earned in humanities and fine arts, together with extensive literature studies. The theses in literature all dealt with fantasy and the final, (yet) unfinished, major thesis concerns both science fiction and fantasy.

After some years in the computer business, the author studied business administration in his spare time to gain a further understanding of business in general. His master's thesis was a study of the Linux industry (Rosén, 2002). This study was based on the industry analysis by Porter (1980), and the result was a surprise. In the terms of the framework provided by Porter, the Linux industry was unlikely to survive, and the pressure from rivals, new entrants and substitutes would make existing companies extremely exposed. Yet, despite the findings, Red Hat was operating successfully. This result spurred a further interest for the author's research in the business logic of the OSS industry.

2.4.4. Experiences

Books and computers played important roles for decisions in the author’s life, but the latter also affected the author’s employment. Together with his friends, the author joined the com-

---

\(^5\) There is an excellent article on BASIC in Wikipedia (2011a).
\(^6\) Wikipedia (2011b) has more information on assembler language.
\(^7\) More information about Lysator can be found at: Lysator (2009).
\(^8\) For more information, see: NannyMUD (2005), and for more information on LPC, see: LPC (1998).
puter societies of the university. There were two, Lysator and CTRL-C.⁹ In these societies, the author was introduced to UNIX, OpenVMS and Linux, but unfortunately missed the opportunity to meet Linus Torvalds at the Lysator 20-year anniversary in 1993.¹⁰ However, these computer societies played an important role in the author’s career. In 1997, the author was given the position of regional manager for a start-up company, Gladius AB. The company, founded by some members of CTRL-C, was formed to develop safe e-business systems, based on OpenVMS. Moreover, the company also introduced a solution based on Linux. The CD-ROM with the Linux distribution was purchased from Signum Support AB, a company founded by members of Lysator.

After two years at Gladius AB, the author took a position as a business unit director at Cendio Systems AB, which had recently changed its name from Signum Support AB. Again, the position was offered based on recommendations from members of both computer societies. At Cendio Systems AB, the author was to launch a new business unit, Systems Solution, which offered solutions based on Linux and OSS. In order to provide customers with hardware certified for Linux, the firm became a reseller for IBM and SGI. At this time, Cendio Systems AB was the only Swedish company providing full Linux support for IBM. However, most of the firm’s revenues were from telecom. When that market failed in 2002, Cendio Systems AB was forced to transform itself, from a mixed business firm to a software vendor.

The management team, consisting of one of the founders, the CEO and the author, finished the restructuring in November 2002, when the new product, ThinLinc, was introduced and the company changed its name to Cendio AB. At that time, the author was considered an expert on the business side of OSS in the specialist press, and became a columnist in Computer Sweden between 2003 and 2004.¹¹

Through his connections, the author was offered a position as an industrial PhD candidate at Linköping University in 2004. The subject of his research was to study OSS companies’ business models in order to find viable concepts. Cendio AB was identified as one of the companies to be studied. In 2005, the author left Cendio AB to work as a full-time PhD candidate.

---

⁹ For more information on CTRL-C, see: CTRL-C (2004).
¹⁰ All the important events at Lysator have been presented in a timeline, available here: Lysator (2009).
¹¹ These chronicles are still available on Computer Sweden’s website (in Swedish), but are somewhat hard to find. One example is: Rosén, (2004).
3. Business models – a background

When Michael Rappa began his online lecture with the words, “Business models are perhaps the most discussed and least understood aspect of the Web”, the concept of the business model was in its infancy (2000). In his online draft, Stähler (2002) argued, “A business model is something very simple. It is a model of an existing business or a planned [sic] future business. A model is always a simplification of the complex reality.”

Now, almost a decade later, the discussion and confusion is still rampant and the “normal” usage among businesspersons seems not to have changed during that time, if you compare the findings of Linder and Cantrell (2000). Even modern articles sometimes dismiss business models as a description of how a business achieves its revenues (e.g. Pauwels and Weiss, 2008).12

However, after the bubble burst of the dot-com companies, the concept fell out of fashion (Magretta, 2002). Interestingly, there are indications that the use of the term "business models" in business journals seemed to follow the index of the NASDAQ stock exchange (Osterwalder, Pigneur and Tucci, 2005).

Academic interest in business models is still high. The strategic management journal *Long Range Planning* devoted Issues 2-3 in 2010 to business models, stating the value of the concept based on its “exceptional importance to managers, and the choice of business model is typically seen as a key component of organisational success” (Baden-Fuller, Demil, Lecoq and MacMillan, 2010a).

An important question is how to define business models. Amit and Zott define it as “the business model depicts the design of transaction content, structure, and governance as to create value through the exploitation of business opportunities” (2001, p. 494 f.). Afuah defines it as a “framework for making money” (2003, p. 2), while Magretta views it as “stories that explain how enterprises work” (2002, p. 87). Chesbrough and Rosenbloom say the concept “provides a coherent framework that takes technological characteristics and potentials as inputs, and converts them through customers and markets into economic outputs” (2002, p. 532).

It is plain to see that the definitions vary, but there are also differences regarding their uses. While some see business models as *descriptions* of the way firms conduct their business (Magretta, 2002) or *simple formulas of business concepts* (Kraemer, Dedrick and Yamashiro, 2000; Rappa, 2000), others use the concept as an *analytical tool* to investigate organizations (Amit and Zott, 2001; Chesbrough and Rosenbloom, 2002; Hedman and Kalling, 2002; Afuah, 2003).

The concept of the business model can be traced to *information and communications technology* (ICT) and the dawn of e-commerce (Timmers, 1998). The majority of the early articles are focused to both explore and explain the phenomenon of e-commerce (Timmers, 1998; Rappa 2000; Papakiriakopoulos, Poulymenakou, Doukidis, 2001; McCann and Lyytinen, 2002; Gebauer and Ginsburg, 2003; Li and Yousept, 2004; Bambury, 2006). In addition, they (more or less) normatively recommend business models or strategy for the avid entrepreneur (Mahadevan, 2000; Linder and Cantrell, 2000; Gordijn and Akkermans 2001a; 2001b; 2003; Petrovic, Kittl, Teksten, 2001).

The business model concept may have sprung from dissatisfaction with existing strategy theories, which could not by themselves explain how value is created when the business was

---

12. This interesting article studies the effects on income and subscribers when an online service went from “free to fee”. Although revenues are an intrinsic part of business models, the authors’ use of the term mimics how the term is used in the IT industry.
exercised on the Internet (Amit and Zott, 2001). Another view of the business model concept is as an answer to the fragmentation of the strategy theory, in which there is empirical data that indicates that modern firms formulate business models rather than create strategic plans (Hedman and Kalling, 2002a).

The term “business model” is older; one academic article uses it as early as 1957 (Bellman, Clark, Craft, Malcolm and Ricciardi, 1957) 13. Its wider use and associated theories appeared much later, and the acceptance of the concept could be connected to the falling transaction cost due to the use of the Internet as information carrier (Osterwalder, Pigneur and Tucci, 2005).

In the literature, there are discussions whether business models should be considered business processes (Gordijn, Akkermans and Vliet, 2000), while at the same time modeling tools have been introduced that can be used to shape predominately Internet-based business (Gordijn and Akkermans, 2001a; 2001b; 2003; Osterwalder and Pigneur, 2002).

Even for those organizations that do not formulate their own business model, the models can still be utilized by external parties to understand how these organizations operate. One use of the business model in the academic world is as an analytical tool, where it combines diverse management and strategy theories, as for example by Hamel (2000), Amit and Zott (2001), Chesbrough and Rosenbloom (2002), Hedman and Kalling (2002a; 2002b) and Afuah (2002).

Finally, there is also a string of research, which concerns the actual business model research, which aims to find delimiting factors among the various examples of business models (Alt and Zimmerman, 2001; Pateli and Giaglis, 2003; 2004; Osterwalder et al., 2005). While Alt and Zimmermann (2001) presented an overview for a special edition of Electronic Market, covering the different aspects of the term “business models”, Pateli and Giaglis (2004) used eight different subdomains, while Osterwalder et al. (2005) used three different levels when they classified business models.

### 3.1. Business model categories

This thesis seeks to add another way to discern between different classes of business models, based on how they are utilized. The reason for this is that the term or even the concept is described and used in a disparate way, as illustrated in the chart shown in Figure 2.

---

13 The business model in Bellman et al. (1957) is really a computer-aided game simulation as part of an educational tool. As a side note, even if the word “business model” only is in the title, Gardner (1960) is also discussing computer-aided simulation games.
In order to create a visual and conceptually understandable mapping, the chart is divided into four dimensions paired into “opposites.” These opposites address four aspects of the term “business model.” Generic business models aim to create or describe such models on a larger scale, in order to investigate or categorize business industry-wide. Specific business models are more often concerned with one business or industry, and intend to describe the particulars of that business or industry. Descriptive business models merely describe business models, without judging them, while normative business models promote some models over others.

Accordingly, Figure 1 illustrates four different categories of business models that should cover most business models: business model concepts, business model descriptions, business modeling, and “successful” business models. The reason for marking the “successful” with quotes refers to the observation that these descriptions usually have a best-before date. Business models do not last forever, or even a very long time.

Please note that that there are some concepts that bridge the borders of one or more of the categories, while the majority of the business models should fit in one box.

### 3.1.1. Business model concepts

The generic/descriptive Business Model Concept encompasses not only those trying to create generic business models (Linder and Cantrell, 2000; Chesbrough and Rosenbloom, 2002; Hedman and Kalling, 2003; Morris, Schindehutte and Allen, 2003; Osterwalder et al., 2005; Zott and Amit, 2010), but also those making use of business models as an analytical tool (Amit and Zott, 2001; Afuah, 2003; Pateli and Giaglis, 2004).

Characteristically, these business models do not promote certain models as superior to others, but instead make use of cases as illustrations for their concepts or principles. Despite the use of cases from different industries or regions, these models aspire to be generic, rather than specific.
3.1.1.1. Generic business models

Business and change models

In a research paper for the global consulting company Accenture, Linder and Cantrell (2000) summarized how the term was used in business. One conclusion from the article was that a major factor for the confusion of the term “business model” was because different authors meant different things when using the term. This was often connected to the offering, for example revenue and pricing models, but also concerned process models and value propositions.

In order to identify the business model in an organization, they proposed four steps: identify revenues, find the value propositions, categorize the delivery and funding models, and find the assets, capabilities and relationships within and outside the organization. Based on these factors, a number of business models could be identified, and the paper presents an extensive list of different variants.

The authors also discussed how changes are made to business models, and identified four types of models for change: realization models, renewal models, extension models, and journey models. The realization model concerns maximizing the existing model, for example by expanding the geographical market, and constitutes the least change to the existing business model. Similarly, the renewal model consists of creating new offerings to keep ahead of the competition, while retaining the existing business model. On the other hand, the extension model offers moderate changes to the existing business model, since it adds to the present concept, where the company is stretching the current model to cover new markets, etc. The most severe changes to the existing business models are brought from the journey model, where the organization is actively moving towards a new business model.

This paper was early in the business literature (2000), and it has affected later business model studies. However, the change models have seldom been discussed per se, even if Hedman and Kalling (2003) add a longitudinal aspect to their framework (see below).

Business models for innovation

Chesbrough and Rosenbloom (2002) expanded the business model concept to encompass innovation, with examples from ventures that originated from Xerox’s research facility in Palo Alto (PARC). Their business model concept encompassed six attributes: value proposition, market segment, value chain, cost structure and profit potential, value network and competitive strategy. These attributes also have two additional functions, to justify the investment and to define ways to scale up the business.

Chesbrough and Rosenbloom also identified three differences between business models and strategy. Business models deal with creating value for the customer and the business in an environment that has limited knowledge of the firm, its customers and third parties. By comparison, strategy concerns threats from competitors and creating value for shareholders, which are based on analyzing assumingly richer information.

Business models in Information Systems

Hedman and Kalling (2003) focus on business models in the Information Systems (IS) setting. Their concept is based on the three main strategy perspectives, industrial organization (I/O), resource-based view (RBV) and strategy process. From these, they integrate concepts from all three perspectives, namely customers, competitors, the offering, the activities and organization, resources, factor and production inputs, and longitudinal process. These components are divided into four levels: market (I/O), offering (generic strategy), activity and organizational (I/O) and resources (RBV).
One aspect of this particular concept is that a change of the business model is also covered, and the authors conclude that the changes can be both exogenous and endogenous, and that all changes to one component will have ramifications throughout the whole model. Even if the purpose was to propose a model to study IS specifically, the proposed concept can be used in other industries.

A unified business model perspective

Morris et al. (2003) are working towards a unified perspective for business models, and thus creating a model for entrepreneurs, focusing on the first stages of a venture. The authors have composed their model based on earlier work, and their composite model consists of three levels, namely foundation, proprietary, and rules. The foundation level addresses the six key factors that will be the basis for the business model: value creation, customers/stakeholders, competence, competitive positioning, revenues, and investor factors. The next level, the proprietary level, depends on the entrepreneurs’ ability to compose unique solutions to one or more of the factors in the foundation level. This ability is what ultimately makes or breaks the success of a venture. This part of the business model may even be considered as intellectual property.14 The final level, the rules level, sets the guiding rules based on the earlier levels and is vital for the execution of the business model as a whole.

Clarifying the business model concept

A major contribution to business models came from Osterwalder et al. (2005), who took it upon themselves to clarify the concept of business models. In order to achieve this, the authors made an extensive search for the term “business model” in scholarly reviewed journals between 1990 and 2003, which showed that the popularity of the term rapidly increased at the end of the 1990s, coinciding with the rise of dot-com business.

The authors then classified the business models into two levels, conceptual and instance, and further classified the business models into three categories, business model concepts, business model types and instanced business models that are tied to a specific company. Business model concepts are generic and aim to describe all types of business models, and are focused to identify elements that belong to business models. Business model types concern taxonomies for business models, and study similarities between their characteristics. Finally, the instanced business models aim to describe individual, real world companies’ business models.

In addition, Osterwalder and the co-authors also found a clear evolution of the business model concept and identified five, chronological, activities. The first activity was to classify and define business models, while the second was to list business model components. The third activity was mainly to describe business model elements, while the fourth started to model and test the business model elements. The final, on-going fifth activity is to apply business model concepts.

The authors also identified nine “building blocks” of business models: value proposition, target customer, distribution channel, relationship, value configuration, core competency, partner network, cost structure and revenue model. These building blocks, in turn, address four business domains: product, customer interface, infrastructure management and financial aspects.

Moreover, five different functions for business model concepts were found: understanding and sharing, analyzing, managing, prospects and patenting. While the first four functions

14 Some companies, like Amazon.com, tried to patent some of their proprietary business model innovations, like the “one-click-purchase.” However, these business method patents have come under fire due to the hasty filings, which may be hurtful (harmful?) for many innovative companies (Lesavich, 2001).
concern different utilization of business models, the final function only concerns patenting of 
business. The final part of the article presents eight propositions on how the business model 
concept can be used within the IS domain.

George and Bock (2011) also attempted to clarify and, ultimately, propose an integrated 
approach to business models to enable entrepreneurs and researchers to “unlock entrepreneurial 
processes, evaluate form configuration effects, and explain and predict entrepreneurial out-
comes” (George and Bock, 2011 p. 107). Their work was based both on a literature review 
and surveys directed to active executives. The outcome was that there is a distinct difference  
between strategy and business models. Strategy concerns a “dynamic set of initiatives, activi-
ties and processes” (p. 102), while the business model is a “static configuration of organiza-
tional elements and activity characteristics” (ibid.). They also stated that if a firm implements 
a business model, it may generate changes in the organization, but the business model itself  
cannot be seen as a recipe or description for change in general. They concluded: “Business 
models are opportunity-centric, while strategy is competitor or environment-centric” (ibid.).  
The study shows that there is a distinct difference between business models and strategy. 
Even if the authors’ call for a more integrated approach for business models research is likely 
to be unheard, their research also shows the importance of business models and strategy living 
side-by-side in academic research.

3.1.1.2. Business model as analytical tools

Amit and Zott (2001) found through their study of a number of American and European e-

business that none of the existing strategic or entrepreneurial management theories could sing-
le-handedly fully explain how value was created in e-business. The solution was to con-
struct a business model that used several of the theories in order to explore e-business. Their 
model implemented value chain analysis, Schumpeterian innovation theory, RBV, strategic 

t network theory, and transaction cost economy (TCE).

In addition, Amit and Zott clearly differentiated the business model from plainly generating 
revenues. They set up a separate definition for this, called revenue model:

Definition: A revenue model refers to the specific modes in which a business model enables revenue


The main difference between the two is that the business model addresses value creation 
while the revenue model concerns value appropriation. The business model deals with “the 
content, structure, and governance of transactions designed so as to create value through the 
exploitation of business opportunities” (p.511).

When it comes to the factors that determine the business model, Amit and Zott concluded 
that there are four factors to reckon with: efficiency, complementarities, lock-in, and novelty


Efficiency stands for transaction efficiency and shows how effectively an e-business deliv-
ers information or goods to its customers. The term “complementarities” stands for the bun-
dling of products or services that are together worth more than separately. “Lock-in” was 
coined to describe how customers or strategic partners are motivated to repeat transactions or 
prolong and/or improve the association. Finally, novelty stands for the innovations created by 
the firm concerning products, services, distribution, or marketing (ibid.).

Shi and Manning (2009) gave the business model as an analytical tool a new direction, pre-
senting a framework for understanding and evaluating risks connected with business models.

15 The ability to patent business models is not available in all parts of the world, so the examples in general con-
cern US patents.

16 E-business is defined as “business conducted over the Internet” (Amit and Zott, 2001, p. 493).
The framework was created from four elements or models, *exchange*, *organizational*, *resource*, and *financial*, which were based on earlier work on business models. The author considered three levels of risk for business models: first, the *business model element level*, where the risk can exist in each element; second, risks in the level of *between-element relationships*, where misfits between two elements could be disastrous; and third, risks to the *whole system level*, where the business model as a whole could face risks.

The main reason for developing this framework was to create a risk management tool with higher granularity, and one designed with business models in mind. This framework was conceived to contrast with earlier models.

Whereas this framework is a business model concept, as well as an analytical tool, it is also aimed at professionals and thus would touch both the “business modeling” and “successful business model” borders in the chart above. It would be interesting to investigate if this tool has been used to analyze business models in a company or organization.

**Activity-based business models**

Zott and Amit (2007) introduced a novel concept where the activities of the firm were studied, which included the design of business models. The focus of this concept was the “boundary-spanning” activities, i.e. how the firm interacts with its environment. The ways these interactions are designed are equal to business model design. Two dimensions of business model design were identified, *efficiency-centered* and *novelty-centered*. The first concerns reducing transaction costs for all stakeholders, while the latter refers to new ways of conducting business (or economic exchanges). These two dimensions are not exclusive and can be present in any business model design.

Based on their theories, the authors propose six hypotheses connected to the two dimensions and firm performance in environments with high or low resources (i.e. financial capital). These were tested on 190 firms that went public in Europe and the United States between 1996 and 2000. The most important finding was that the firms using business models were efficient, without reference to environmental conditions. In addition, firms with a novelty-based business model were shown not only to innovate through internal changes, but also through external stakeholders.

Expanding on this theme, Zott and Amit (2008) tested their concept differently by analyzing the fit between product market strategy and the business model. Through a sampling consisting of 170 European and American companies that went public between 1996 and 2000, the fit was tested through statistical computer models. The result showed that both market strategies and business models affected the firms’ market values. Moreover, the combination of market strategy and business model was complimentary rather than independent.

In 2010, Zott and Amit summarized their activity systems for business models. Activity systems describe how firms do business, and by looking at the activities, the authors identified two different sets of parameters for business models, *design elements* and *design themes*.

There are three design elements, *content*, *structure* and *governance*, where the first refers to the activities of the firm, i.e. what the firm actually does. The structure denotes how the activities are *connected* with each other and how *important* they are for the business model. Finally, the governance concerns *who* is performing the activities.

Design themes, which are configurations of design elements, are another way to study activity systems. The authors introduce the NICE acronym, which consists of *Novelty*, *lock-In*, *Complementarities* and *Efficiency*. Novelty concerns new content, or new ways to handle structure and governance. Lock-in refers to retaining customers, where the activity system can contain switching costs, or other ways to increase customer retention within the content, structure or governance of the activity system. Complementarities come from bundling activi-
ties when these add more value than being separate. Finally, efficiency concerns reducing transaction costs through more efficient activity systems.

This concept borders the business modeling part of the chart (see below), since it can be used to both describe and analyze business models and create new ones.

**Analyzing E-business models**

Pateli and Giaglis (2003; 2004) developed a research framework to analyze e-business models. When reviewing the existing literature on business models they identified five objectives for studying them: understanding key elements in specific business domains, communicating business models to various stakeholders, designing information and communication systems, experimenting with business concepts, and changing or improving the existing business model.

The authors found eight sub-domains of business model research: definitions, components, taxonomies, conceptual models, design methods and tools, adoption factors, evaluation models and change methodologies. Based on the findings the authors identified a number of challenges to creating a conceptual framework, but they also concluded that the research field for business models had matured from mere descriptions towards providing tools and insights for business managers. However, the authors also saw a need for the researchers to develop a common language, akin to Linder and Cantrell (2000), in order to create more elaborate business model conceptualisms.

Other authors also looked at e-business models. For example, McGann and Lyytinen (2002) proposed a framework for organizing and analyzing elements of e-business. Their concept was an extension of Timmers (1998) and Kraemer et al. (2000), and introduced the EAF (eBusiness Analysis Framework), which contained ETI (Electronic Trading Infrastructure) components. This system was cumbersome and based on an inflexible view of the business model, which made it susceptible to rapid changes.

**3.1.1.3. Summary of generic business models**

The articles and books discussed above have several things in common aside from the term "business models". They all concern the need to see beyond the older strategy and management theories. Many of the articles, in particular the older ones, are related to fast-moving industries, mostly the ICT industry, which may signal that the business conditions in these industries are destined to render earlier theories of the firm incorrect or incomplete.

Whether or not this is view is correct, the business model concept offers a different way to study the increasingly complex world of organizations. By marrying theories and concepts that the respective proponents consider incompatible, generic business models may be the bridge to consolidate different views of the firm and thus further the understanding of how firms conduct their everyday business.

**3.1.2. Business modeling**

In comparison to business model concepts, generic/normative business models are concerned with actual concepts that help companies to create their own business models. Most of the concepts were designed for e-business around 2001 (e.g. Gordijn and Akkermans; 2001a; 2001b; Petrovic et al., 2001; Papakiriakopulous et al., 2001; Osterwalder and Pigneur, 2002; Hayes and Finnegan, 2003). After the bust of the “dot-com era”, there were fewer examples of business modeling; Demil and Lecoq (2010), however, present a more recent effort.

**E3-value** (Gordijn and Akkermans, 2001a; 2001b; 2003) was designed with e-business in mind, which could be implemented by traditional business to investigate the potential profitability of moving into the Internet. The model is based on economic value and uses Use Case
Maps (UCM), a lightweight scenario technique to construct their ontology. The concept was complex to cover many different types of businesses.

Petrovic et al. (2001) set out to create business models for e-business based on an interdisciplinary methodology from Evolaris. Evolaris is an Austrian joint venture with researchers and major corporations. The concept is based on seven sub-models, value, resource, production, revenue, capital, market, and customer relations, which in turn consist of three additional sub-models, distribution, marketing and service. These models are used in a methodology, which creates a business model in six steps. The idea is to present a generic and normative approach for companies to develop e-business models.

Papakiriakopulous et al. (2001) have created an analytical framework for business models in order to analyze and develop e-business models, to help traditional business to take the step into the electronic market. The framework consists of four major elements: coordination, cooperation – competition, customer value and core competence. The framework is found on the border between the normative and descriptive boundaries on Figure 2, since the framework is descriptive, but the implications are normative.

Osterwalder and Pigneur (2002) created their ontology for e-business based on four elements: product innovation, customer relationship, infrastructure management and financials. These, in turn, have several underlying concepts that together make the foundation for a business model. This concept was designed with computer-aided design environments in mind, where the different elements could be used to design and test business models.

Hayes and Finnegan (2003) created the prerequisites model, which functions as a framework for businesses to decide which electronic business model is the best fit for the particular firm. This framework was based on earlier articles, for example Timmers (1998), and the model was operationalized by checklists where the firm could rate different aspects of the intended business and narrow down the number of acceptable business models. One issue with this concept was that new or hybrid versions of the listed business models could not be managed, making the framework inflexible.

Demil and Lecocq (2010) also proposed a business modeling framework, called RCOV, which stands for resources and competences (RC), organization (O) and value propositions (V). This concept aims to create a feasible means to establish both the static and the transformational approaches to business models. The RCOV concept addresses both approaches by focusing on the interactions within and between the elements. This concept of business modeling hovers just beneath the line of the business model concept in the chart; it is that generic in form. However, since the utilization of the concept is presented as a tool for practitioners to shape their business, it is also normative.

### 3.1.2.1. Summary of business modeling

The generic yet normative approach to the business model had its heyday in the dot-com era, when confusion regarding business models was at its peak. The models were designed to use computer programs, where would-be business managers could design and test different business models before they were launched “in the wild”. However, the normative approach may limit the usefulness over time. Perhaps this is why very few modeling concepts have been presented since 2002.

All of these modeling concepts use different approaches, employing different aspects of value creation as the common denominator. The latest concept from 2010 addresses the important issue of change, which is increasingly important in the fast-moving industries. Models like RCOV (Demil and Lecocq, 2010) are better suited to handle this issue, and perhaps this concept will be adapted into software.
3.1.3. Business model descriptions

The descriptive/specific Business Model Descriptions cover a wide array of taxonomies (Timmers, 1998; Bambury, 1998; Rappa, 2000; 2001; 2004), but also survey regional or specific markets (Lechner and Hummel, 2002; Krueger, van der Beek and Swatman, 2004; Li and Yousept, 2004.)

3.1.3.1. Taxonomies

Articles covering different types of business models became common in the beginning of the Dot-com era at the end of the millennium. Timmers (1998), who in his article defined the new concept as “electronic commerce”, attempted to provide a framework to classify business models in this emerging market. The definition of the business model included three parts: an architecture for the offering and a description of the actors and their roles; a description of the benefits for said actors; and a description on how revenues are collected (Timmers, 1998, p. 4). Based on this definition, Timmers identified eleven business models: e-shop, e-procurement, e-auction, e-mail, 3rd-party marketplace, virtual communities, value chain service provider, value chain integrator, collaborative platforms, and information brokers. These business models have been classified into two dimensions, functional integration and degree of innovation, where the e-shop is the least innovative model with the least functional integration. On the other hand, the most innovative and model and the one that integrated multiple functions was the value chain integrator (1998, p. 7).

Bambury (1998) adapted a different taxonomy based on two categories, transplanted real-world business models and native Internet business models. He found eight real-world business models that had been adapted to the Internet, mail order, advertising-based, subscription, free trial, direct marketing, real estate, incentive scheme, business to business, as well as discussed combinations of these models. Regarding native Internet business models, six were identified: library, freeware, information barter, digital products and delivery, access provision, and website hosting and other Internet services.

Bambury had previously warned about personal integrity issues connected with Internet-based commerce and that the reluctance to regulate the trade of personal information will result in various inappropriate activities, such as spam, clickstream surveillance, intrusive marketing and profiling (1998, p. 6). These activities are common today and increasingly associated with criminality. Bambury was also skeptical regarding the exponential growth of Internet business, based on the lack of regulation and protection of the consumer. History would prove him right.

Rappa (2000, 2001) starts his online lectures by stating that business models were the aspects of the Internet discussed but not really understood. The definitions for business models are:

“[…]a business model is the method of doing business by which a company can sustain itself -- that is, generate revenue. The business model spells out how a company makes money by specifying where it is positioned in the value chain” (2000, p. 1).

Based on the definition, online business models are classified into nine basic categories: brokerage, advertising, infomediary, merchant, manufacturer (direct), affiliate, community, subscription, and utility. In addition, under each category a number of examples are given for existing online business models. The online lectures are basically a list of existing business models with a short description about each particular model. Later, Rappa expanded the taxonomy to explain how utility computing, i.e. computers-on-demand (also known as ASP, software as a service or cloud computers) could be used to charge the customers (2004).

Mahadevan (2000) has also provided a taxonomy for business models on the Internet by categorizing different models through their market structure. He identified three emerging mar-
Market structures: portals, market makers, and product/service providers. While portals create communities or provide focal points for users, the market makers create business, such as brokerage, between actors. Finally, the product/service providers interact directly with their customers.

In addition to the market structures, Mahadevan also covers three streams within the business models, value, revenue and logistics, in order to map these streams to the market structures. In essence, Mahadevan is paving the way for a more generic and conceptual way of addressing business models.

3.1.3.2. Specific markets

Lechner and Hummel (2002) studied business models in virtual communities with the music industry, search engines, and price mechanisms as examples. They made use of Timmers’ (1998) definition of business models (see above) and argued that virtual communities have evolved from social phenomena to a business model (p. 43-44).

By looking at how value chains changed in these industries based on impacts from virtual communities, the authors concluded that the (bargaining) powers of these will increase over time.17

Barabba, Huber, Cooke, Pudar, Smith and Paich (2002) described the General Motors OnStar project. This article is really a case description, providing insights (no doubt doctored) about the process of creating a business model for services outside the normal scope of the company. This article hovers near the “Successful” Business Models border in the chart above, since it describes a successful case. However, the scope is also very limited, making it hard for other companies to copy or adapt similar solutions.

Li and Yousept (2004), Kruger et al. (2004) studied specific emerging online markets. The former studied how supermarkets in the United Kingdom extended their market into the Internet. They found six variables that these businesses need to deal with: channel development and integration, business scope, value network, fulfillment models, personalization and differentiation strategies. In order to succeed in the online environment, all these variables need to be addressed.

Krueger et al. (2004) studied how newspapers handled the emergence of online news. They found that despite the lack of success (this was written in the wake of the recession after the Dot-com era), most newspapers were still providing their news freely on the Internet. There were five trends that needed to be addressed to achieve success: identification and branding, combination business models, identifying comparative advantages, technology push, and niche marketing.

The driving forces were the consumers, who were increasingly asking for online content (for free). While the majority of the online news providers offered their content freely, some were charging for content and others were planning to. The Internet provides infinite opportunities for advertising and marketing; at the same time, it also provides for unlimited competition. The authors conclude the article by stating: “The only certainty for the digital content industry is that there is no certainty!” (Krueger et al., 2004, p. 14).

17 However, the authors used the emergence of the mp3 in the music industry, which first led to the emergence of Napster.com and furthermore to peer-to-peer networks like Gnutella. However, the music industry has managed to battle this phenomenon quite successfully, shutting down Napster.com in 2001 and the servant (peer-to-peer network node) LimeWire in 2010.
3.1.3.3. Summary for business model descriptions

The time for mere descriptions of business models has passed. The confusion regarding the term at the turn of the millennium with the emergence of Internet commerce made these taxonomies useful and necessary. However, as Osterwalder et al. (2005) presented, the evolution of business models in academic circles has moved to another phase. It is doubtful that new taxonomies will occur within academia, but there is a plethora in the non-academic press and literature.

3.1.4. Successful business models

Normative/specific successful business models consist of success stories from companies, with one of the best-known examples from Kraemer et al. (2000) who conducted an extensive, deep case study of Dell Computer Corp, which basically consisted of direct contact and built-to-order offerings with corporate customers. Based on this concept, Dell continuously tried to improve its processes in order to boost profitability. This concept was then translated into its online business, which enabled the company to address new corporate customers, and also consumers with similar offerings to the earlier corporate customers. The authors conclude that the direct sales and built-to-order business model is deceptively simple, making the execution complex. The reason for Dell’s success was the model and the execution, which gave the company a competitive advantage.18

The increase of mobile phone usage has created another market where business models are used to describe how firms operate. Macinnes, Moneta, Caraballo and Sarni (2002) describe business models for mobile games. The authors proceed in manner similar to Lechner and Hummel (2002) by adapting Timmers’ (1998) definition and making use of value chains to describe various existing scenarios for mobile games and provide a number of factors that are important for success. However, the authors do not provide specific cases, and the scope is normative and somewhat limited in scope, hence straddling the border between “business model descriptions” and “business modeling” in the chart above.

Similarly, Gebauer and Ginsburg (2003) describe why two American online wine retailers succeeded, providing nine principles for how (any) online retailers should proceed to become successful.

3.1.4.1. Summary of successful business models

The main aim for this category is to influence decision makers by describing success stories. Their normative purpose could be described either as pragmatically informative or as mere marketing for the selected case companies. The usefulness of the advice is limited, and most case companies would be copied, acquired or outperformed by more innovative companies.

However, these cases deliver a backdrop to the business model concept as a whole, and also provide information for the stories Magretta (2002) discusses. It is doubtful that these normative success stories will gain a foothold in the academic literature, even if they are still popular in the press.

---

18 A note on this case is that competitive advantages seldom last, particularly in fast-moving industries; Dell’s competitors, like Hewlett-Packard, adapted the business model together with strong channel sales and managed to outperform Dell in 2006 (Gartner, 2007),
3.2. Strategy and business models

There is one aspect regarding business models that requires a separate section, and it concerns the relationship between strategy and business models. When Porter criticized business models (2001), it was launched from the strategy standpoint. The articles discussed in this section could also be deferred to each part of the chart above, but are discussed in this specific context.

Seddon, Lewis, Freeman and Shanks, (2004) start from Porter’s article (2001) but also add an earlier article (1996) when they summarize “The Harvard School of Strategy”. They argue that the latest incarnation of strategy, as reported in those two articles, very well covers most of what has been attributed to business models. In addition, they compare this new strategy concept with the business model concept from four articles, concluding that business models are “abstract representations of some aspect of a firm’s strategy” (p. 440).

Considering Morris et al. (2002) it is clear where the criticism from Seddon et al. (2004) stems from. Their definition of a business model is more or less a carbon copy of what earlier strategists have attributed to the strategy concept, where the ultimate goal is to gain sustained competitive advantage. Their ultimate goal is to create a “unified perspective” on business models for entrepreneurs, by turning the arguments of Porter (and Seddon) upside down.

Yip (2004) has a different view. His business model contains nine elements in which the company interacts with its environment: value proposition, nature of inputs, how to transform inputs, nature of outputs, vertical scope, horizontal scope, geographic scope, nature of customers and how to organize. He argues that strategies are used to change the business model when changing environments, and exemplifies this with a number of cases. He differs between environments and positions, where the latter can be addressed through “routine strategies” as proposed by Porter (2001). When a firm encounters a new environment, “radical strategies” are needed in order to change the firm’s business model.

Winter (2003) approaches the problem from yet another angle. He disregards the “business model” literature and states that the term “model” is used in an ambiguous way. He makes use of the model as a process where business strategy and business networks are translated into models in order to classify, standardize, and check them for integrity and completeness. In addition, the models suggest that the planning process can be linked to redesign and development in IT systems, in which components can be reused in latter stages.

Even if this approach were not considered to be “business model theories,” as described above, Winter’s approach is still valid in illuminating the ongoing discussion between strategy and business models proponents and due to the fact that he is using ideas and articles from both “sides”.

For Teece (2010), business models address how firms create value for their customers, how revenues are collected and how the revenues are translated into profit. He also discusses what he recognizes to be a lack of theoretical grounding of business models in economic, organizational, strategic and marketing science. The problem, according to Teece, is the simplified view of the world that misses many aspects considered to be important for “real businesses”. For him, “a business model is more generic than a business strategy” (p. 180) and strategies are as important as the business model itself. A solid business model entails a good strategic analysis.

George and Bock (2011) also conclude that there are clear and distinct differences between business models and strategy, where both constructs are concerned with different aspects of how firms conduct their businesses.

Whether business models are part of strategy, or the other way around, it is clear that the view of how firms operate is of importance. Business models are well-suited to handle com-
plex business situations that often beyond the scope of more traditional theories. The reality for firms, especially in high-tech industries, is complex and fast moving. This calls for a “light-footed” approach as well as more generic concepts than strategies. Strategies, on the other hand, can be the defining difference that sets a particular firm apart from its competitors. Surely, there is a need for both business models and strategies.

3.2.1. The need for business models

Is there a need for business models? If Seddon et al. (2004) are correct, the newest incarnation of the strategy term encompasses the business models they investigated. However, given the breadth of the usage of the business model in articles other than the ones investigated, it would seem that their notion is far from the truth. The reason is that strategy is perhaps not as close to being unified as they claim. Even if they refer to the “Harvard School of Strategy,” there are other streams of strategy (Hedman and Kalling, 2003).

(Mansfield and Fourie (2004) state that strategy is concerned with the desired future of the firm, based on sustained competitive advantage, while business models are concerned with creating value for their customers. Mitchell and Coles (2003) present a similar proposition, stating that strategies are plans and actions to utilize a firm’s resources in order to gain a sustained competitive advantage, while business models ask questions about how a company can provide customers with products and services.

These examples only express one side of the usage of business models and strategy. Notwithstanding the normal usage, there is also another side of the business model that significantly differs from strategy as an analytical tool. This stream of research has been continuous from Amit and Zott (2001), Chesbrough and Rosenbloom (2002) through Hedman and Kalling (2002; 2003) to Osterwalder et al. (2005). These articles make use of the business model as a theoretical framework and add theories from different sources in order to analyze firms and companies. Instead of debating whether business models supersede strategy or vice versa, they mix different strategy and management theories in order to create a vessel for research.

3.3. Discussion

This section was designed to show that business models have a wide array of meanings and uses for scholars. However, one important aspect is missing: the use of the term among practitioners. The term “business model” is used in different forums, but in general concerns high-tech industries, such as computers with hardware and software.

Since there is no clear denotation of the term, the connotation has changed over the years. During the “dot-com era” when the term found wider use, having a business model on the Internet could help attract large investments (Krantz, 2000). From a more cynical point of view, these business models were to raise stock prices rather than profits (BBC News, 2001).

Today, the usage of the term is perhaps more straightforward, and also more widespread. Even Swedish municipalities and local governments use the term when describing financing collaboration projects concerning e-government.19

It is probably safe to conclude that there will be a gap between the academics’ and nonprofessionals’ use of any term, and the business model is no exception. Another rather safe assumption is that the term may change further, in concert with changes in the business climate.

19 This usage has been documented in an ongoing project with the organization Sambruk in Sweden. For more information, see: Sambruk, 2009. Municipalities for Joint Development of e-Services - Sambruk. (PDF). Available: <http://www.sambruk.se/inenglish.4.76e8b1c6112f078db498000144852.html> (2009-04-27).
However, there are still pertinent questions regarding the usage of business models for scholars. Are business models mere descriptions on how firms conduct their business? Are business models simply tools for analyzing or creating a business? Is there a need for a unified framework regarding business models? Are business models redundant with respect to strategy? Alternatively, is strategy a part of business models?

These questions are important, and different authors will (as shown above) answer them differently. However, in this thesis, the answers to the first two questions are “yes,” if the business model description covers more than the revenue model. A good business model framework can function both as a description and as analytical tool. An attempt to produce such a framework is presented below. As shown above, there is not one business model concept, but several. Business models can function in different roles in different contexts, as described in the journal *Long Range Planning*:

Business models are not recipes or scientific models or scale and role models, but can play any – or all – of these different roles for different firms and for different purposes: and will often play different roles at the same time. (Baden-Fuller and Morgan, 2010, p. 168).

The third question requires a somewhat longer answer. Business models can be seen as a “sign of the times” due to the incongruent business climate. In addition, social sciences are not exact sciences, like mathematics and physics. Considering the difficulties that scholars have had trying to create a unified model for physics, it is largely impossible to create one theory that encompasses all types of business.

Moreover, each academic institution sports “house gods” for strategy and management that will affect the affiliated researchers. A unified framework, no matter how stringent, is unlikely to cover all relevant theories and situations and still be possible to work with. In fact, the various attempts to create such frameworks in strategy have failed (despite the claims of Seddon *et. al* (2004). Based on these two arguments, a unified business model framework is likely to fail.

The last questions have already been discussed above. The answers depend on how strategy or business models are used and one’s point of view, but also depend on which concept of business model is used. If the business model only concerns how revenues are gathered, business models can definitely be part of strategy. However, if the business model takes a holistic view, it addresses how firms’ functions and strategies are just a part of the business model. Moreover, redundancy also depends on the point of view. Strategies often deal with plans and concepts on how to address a certain area, such as handling competitors and gaining sustainable competitive advantage, or how to position and market the brands and products of the firm. Business models address other aspects, such as organization or resources (e.g. George and Bock, 2011). Business models are more concerned with how and with whom the firm interacts in order to fulfill the strategies (see Mansfield and Fourie, 2004; Mitchell and Coles, 2003). The business model framework in this thesis adheres to the latter perspective, and includes strategy as a part in the framework.

With this said, it may be confusing for practitioners to hear scholars discuss business models in a different way than they perceive the term. This unfortunate confusion could spell trouble for the much-needed relationships between the two domains. Is there a need to differentiate the term between the scholarly research and the practitioners’ reality? In this thesis, the studied business models of the case companies make use of the term: *business model configurations* in order to differentiate one firm from the other. This concept is also similar to the view of George and Bock (2011). This may be one way to minimize confusion between practice and research.
4. Open-source software – a background

This chapter will present the background and concept of free software (FS) and OSS. However, in order to best present this, a brief and basic explanation of the computer industry and its technology and terminology will be provided.

4.1. The computer industry

The computer industry is widely known as one of the fastest-moving industries in the world, rivaled perhaps only by the emerging biotechnology industry (e.g. Wichmann, 2002a; Hedman and Kalling, 2002). The changes are rapid and often profound. This is because in reality, the computer industry consists of not one, but two fast moving sub-industries: those dealing with hardware and software. Even if they are considered conjoined, each of these sub-industries has had its own history of evolution and revolution, each which eventually affects the other (Levy, 2002).

4.1.1. Hardware

The changes in the hardware industry have been somewhat “regulated” by the now famous Moore’s Law from 1975, which stated that “circuit density or capacity of semiconductors doubles every eighteen months or quadruples every three years” (Schaller, 2006). Instead of revolutions, the progress can be seen as an evolution, and this ongoing process is projected to last for many years (Kurzweil, 2001).

4.1.2. Software

For software, the progress has taken more of a revolutionary path, where new technology, processes or even paradigm shifts have altered the industry in a fundamental way. The software industry has lived through events like “the software crisis” in the 1970s, when companies and universities realized that the complexity of software had to be addressed. This created new research fields and “software engineers,” and turned software into something that could carry value (Levy, 2002).

This new industry was based on the fact that any software created was the main intellectual property of the company, and that the revenue came from the software licenses (Mundie, 2001). This kind of software was termed “proprietary software” by those promoting FS (Stallman, 1985).

Another major event was the open-source software revolution at the end of the last millennium, in which major multinational corporations like IBM and HP openly embraced and even praised the power of software created through voluntary actions, also known as OSS (Randell, 1996).

One of the key issues is how to develop software, and there are different philosophies regarding this. Microsoft and other commercial players use the concept of “good enough” software, and release their products with “known bugs”; this means there are known errors in the code, but it should be “good enough” to ship (Bach, 1996).

However, the philosophy of free or open-source software differs. In general, deadlines are not considered important and the software is released (as stable) when most of the (known) errors are corrected. The way that this is done is also discussed between the main “thinkers” behind the movement, Richard M. Stallman and Eric S. Raymond (Stallman, 2001; Raymond, 2001).
Despite the fact that it has its roots in the dawn of computing (Raymond, 2001; Stallman, 2002), most people seem to consider open-source software (OSS) as something relatively new. But prior to that and even today, most people using the Internet were/are not aware that a major part of their traffic to web pages and through name servers is made possible by OSS. The OSS web server software Apache has been the dominant web server software since 1996 (Netcraft, 2011).

Another important issue considers the rights of the software producer and software users. The commercial producers protect their rights through their licenses, only giving the users the right to use the software under certain conditions and thus limiting the usage. The right to use the software is sold through a license, usually for a fee. This type of software license will be referred as to a “proprietary license” henceforth (Pennington, 2009).\(^\text{20}\)

The free and open-source software producers often do the opposite, giving away the rights of the software to the users. Interestingly, both of these practices use the same basic law, the copyright law. Where the proprietary licenses make full use of the law, terminating most of the rights of the user, the OSS license known as the General Public License (GPL) does the opposite, enforcing the rights of the user; therefore, it is often referred to as “copyleft” (Stallman, 2001).

4.2. Hardware and software

Most modern electronic devices could be called “computers”; whether they are washing machines, refrigerators, mainframes, servers, PCs, laptops, or mobile phones, they all have both hardware and software.

The hardware is the physical machine consisting of electronic components like the processor, memory, screen etc. In order to do anything, and to enable us to control the device, the hardware needs software, called drivers, to run.

The drivers contain instructions, explicit commands for the hardware, thus controlling their performance in certain situations. An operating system is bundled software that not only contains drivers, but also software for the user, i.e. the person using the machine, which enables one to manipulate the hardware and to see (or hear) the result on a screen (or speakers). As Figure 3 shows below, there are some differences between personal computers, which typically have only one or a few users, and servers, mainly located in server rooms and “serving” several personal computers.

\(^{20}\) However, it is important to distinguish between commercial and proprietary software. Even if the latter often is both, commercial software can be open and still sold (Pennington, 2009).
Software can be compared to the recipe for a dish (Raymond, 2001). In order to make this analogy, we need to divide the software into two different “states,” the source code and the binary code.

Source code is readable for humans and consists of text and numbers, while binary code only consists of binary data, which is executable and interpreted by the hardware. The source code can be seen as the recipe, which the cook reads or creates in order to make a dish. It is made in a certain language, just as recipes can be made in English, Swedish, or French. However, languages for software are constructed rather than developed over time (as our natural languages). Some examples of programming languages are Java, C, C++, and XML etc.\(^{21}\)

Just as a recipe needs to be mixed and cooked in order to become a dish, the source code needs to be converted into binary code. Some languages use certain software that interprets the source code, for example Basic, while others like C and C++ use special software called compilers that generate the binary code.\(^{22}\) To return to the food analogy, compiling is the mixing and cooking; next, the software needs to be executed in the system, or eaten as a dish. The performance of the software is dependent on the quality of the source code and the ability of the compiler, just as dishes need a good chef and quality ingredients.

### 4.3. Licenses

Almost all software comes with a license. This is because all software is protected by copyright legislation.\(^{23}\) The copyright is created to protect the creator and means that a non-licensed user has no rights to use the software. A license grants the user certain rights, togeth-

---

\(^{21}\) The number of programming languages increases quickly; for an extensive and updated list of programming languages, see Wikipedia, 2011c.

\(^{22}\) The GCC compiler (formerly GNU C Compiler, now GNU Compiler Collection) was one of Richard Stallman’s projects when he started the Free Software Movement (Stallman, 2002).

\(^{23}\) In some countries such as the USA, some software can also be patented. In general, however, software is protected through copyright laws.
er with a disclaimer, ensuring that the copyright owner will be free of damages incurred by
the software.

There are different types of licenses. Some are proprietary licenses, often called the End
User License Agreement (EULA). This type of license normally specifies the rights of one
software or company. However, Microsoft, for example, has a large number of licenses.
Another type of license is the non-commercial license, called freeware, shareware, public
domain, FS and OSS. These licenses are generic and are used for many programs.

The terms of the licenses differ with respect to granting rights to the users. The public
domain gives the user all rights to the software, including changing the license to a commercial
one. For example, Apple’s operating system Mac OS X is partly based on UNIX from the
public domain (Apple, 2011).

On the other hand, proprietary licenses grant very few rights and typically charge the user a
fee for the rights entitled. Most commercial software today makes use of proprietary licenses.

Freeware and shareware are licenses that lie between the public domain and proprietary li-
censes. Freeware are programs free to use, but give no additional rights to the user. Shareware
comes in a variety of licenses; some offer limited functionality, while others have a time limit
on free use. Shareware is usually used as a “teaser” or promotion for proprietary software
(Knopf and Ford, 2000).

Another difference is how the software is distributed; proprietary licenses, shareware, and
freeware normally only distribute the executable, binary code, while OSS often comes with
both the source code and the binary code. For comparison, Table 1 shows the differences be-
tween the mentioned licenses:

Table 1: Different types of software licenses.

<table>
<thead>
<tr>
<th>Type of license</th>
<th>Can be commercialized</th>
<th>Source code included</th>
<th>Binary code included</th>
<th>Charges fee for use</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public domain</td>
<td>X</td>
<td></td>
<td></td>
<td>-</td>
</tr>
<tr>
<td>Open-source software</td>
<td>X&lt;sup&gt;24&lt;/sup&gt;</td>
<td>X</td>
<td>X</td>
<td>-</td>
</tr>
<tr>
<td>Free software</td>
<td>-</td>
<td>X</td>
<td>X</td>
<td>-</td>
</tr>
<tr>
<td>Freeware</td>
<td>-</td>
<td>-</td>
<td>X</td>
<td>-</td>
</tr>
<tr>
<td>Shareware</td>
<td>-</td>
<td>-</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Proprietary licenses</td>
<td>-</td>
<td>-</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

The term OSS refers to how a program is licensed in a certain way to guarantee that the user
gets extra benefits, i.e. the rights to not only use the software, but also to read the source
code, change it and redistribute it. By giving the users these rights, the idea is that the soft-
ware evolves at great speed (Open Source Initiative, 2011a). In addition, most licenses used
for FS are valid OSS licenses. In fact, the most commonly used OSS license is the General
Public License (GPL), which was created for FS by Richard Stallman (Wheeler, 2005; Stall-
man 2002).

4.4. Free software or open-source software

FS and OSS are different terms describing the same movement, but from different points of
view. Free software originates from Richard M. Stallman, who created the concept in the

---

<sup>24</sup> Some of the licenses allow licensing fees.

The “movement” is about software that was developed by a large number of developers, and it is free for anyone to use, develop, and distribute. Even though the concept of free software has existed since the early days of the computer industry, companies have transformed it by creating business based on it (Levy, 2002). The interest in the movement has been increasing in the academic world, and the research has been centered on the writings of Stallman (2002) and Raymond (2001).25

4.4.1. Free software, Richard M. Stallman

Stallman came to the Artificial Intelligence Lab at MIT in 1971 and embraced the software-sharing culture. In his book Free Software, Free Society (2002), he describes his time at MIT and how his GNU (GNU’s Not Unix!) project started. In 1984, he left MIT and started to work on GNU software, wrote the GNU Manifesto (Stallman, 1985), and in 1985 created the Free Software Foundation. The heart of his work lies in the GNU General Public License (GPL), which Stallman first released in 1989 (Stallman, 2002).26 The concept of the GPL has been called “copyleft,” since it negates many of the rights proposed by American Copyright laws:

My work on free software is motivated by an idealistic goal: spreading freedom and cooperation. I want to encourage free software to spread, replacing proprietary software that forbids cooperation, and thus make our society better. That’s the basic reason why the GNU General Public License is written as it is – as a copyleft (Stallman, 2002, p. 91)

Stallman takes a “moral and ideological” point of view on the creation and dispersion of source code. He is a staunch opponent to closing software, and an opponent of OSS. This quote illustrates his point of view:

For the Open Source Movement, non-free software is a suboptimal solution. For the Free Software movement, non-free software is a social problem and free software is the solution (Stallman, 2002, p. 55).

Stallman’s stance is based on the concept “Free as in Freedom” – a freedom which is nullified by the use of patents and copyrights. He argues that the patent system has failed, since the original concept was to promote innovation, but now it is used to stop them. He considers these practices unethical, even “anti-social”. However, he is not opposed to making business based on software, but states that the business should come from services, rather than closed proprietary software licenses (Stallman, 2002).

4.4.2. Open-source software, Eric S. Raymond

Eric S. Raymond, who was also a part of the free software movement, saw other important issues than Stallman did, issues based on a practical sense (Raymond, 2001). What Raymond observed was the new development process that Linus Torvalds introduced for the Linux Kernel project:

25 Both books contain shorter essays that have also been published on the Internet. Raymond’s essays have also been publicly revisited continuously. The content presented here is mainly based on the published books.
26 The second version of GPL was released in 1991 and the forthcoming third version was formalized in 2007 (Free Software Foundation, 2010).
Linus Torvalds's style of development - release early and often, delegate everything you can, be open to the point of promiscuity - came as a surprise. No quiet, reverent cathedral-building here - rather, the Linux community seemed to resemble a great babbling bazaar of differing agendas and approaches (aptly symbolized by the Linux archive sites, who'd take submissions from anyone) out of which a coherent and stable system could seemingly emerge only by a succession of miracles. The fact that this bazaar style seemed to work, and work well, came as a distinct shock (Raymond, 2001, p. 21).

Raymond also envisioned a new way of developing software – where the input from a multitude of both users and developers was a powerful and effective way to create high quality programs. He also coined a new “law” for software development, affectionately dubbed “Linus’ Law”: “Given enough eyeballs, all bugs are shallow” (Raymond, 2001 p. 30).

4.4.2.1. Commercializing “free software”

In 1998, Netscape decided to open up the source code for its web browser, and Raymond (and other community leaders) felt that the label “free software” was troublesome to use when talking to commercial actors:

Specifically, we have a problem with the term "free software,” itself, not the concept. I've become convinced that the term has to go. The problem with it is twofold. First, it's confusing; the term "free" is very ambiguous (something the Free Software Foundation's propaganda has to wrestle with constantly). Does "free" mean "no money charged?” or does it mean "free to be modified by anyone," or something else? Second, the term makes a lot of corporate types nervous. While this does not intrinsically bother me in the least, we now have a pragmatic interest in converting these people rather than thumbing our noses at them. There's now a chance we can make serious gains in the mainstream business world without compromising our ideals and commitment to technical excellence - so it's time to reposition. We need a new and better label (Raymond, 1998).

Many prominent community leaders at the time lauded the new suggested label, “open source,” even if founder Richard Stallman was against it (Raymond, 1998).

4.4.3. Differences between Stallman and Raymond

As the founder of the free software movement, Stallman created certain ideological standpoints. His point of view is always that of freedom, freedom for the creators and users of software. He is opposed to software patents and all types of licenses that shackle the user (and programmer). Stallman has chosen to take the role of the moral and ideological bastion when it comes to software.

Raymond, on the other hand, is not that opposed to many of Stallman’s ideas and points of view. However, what Raymond proposes is that the software created by the community is better than the traditional proprietary software, and through this, companies would benefit by working with OSS.

This pragmatic view of OSS is the reason for this research. Even if the author agrees with many points made by Stallman, the pragmatic view on OSS is central for this thesis. Henceforth, if not stated otherwise, the term OSS will be used to describe the whole movement, but with a bias towards the pragmatic view.

4.5. OSS research streams

Both Stallman (1985, 2002) and Raymond (1998, 2001) have heavily influenced most research concerning OSS. Nevertheless, the author has identified three distinctive research streams for OSS, motivation of developers, user innovation, and the economics of OSS. The latter addresses how companies use OSS, both as users, but also concerns developing business from OSS, including business models.

One of the largest studies conducted on OSS was the FLOSS report. The project, funded by the European Commission, made its final report in 2002. It consisted of seven reports and was
divided into six main parts, investigating OSS developers, governmental agencies and companies in Europe (Ghosh, et al., 2002a, Ghosh Ghosh, Krieger, Gloot and Robles, 2002b; Ghosh, Ghosh, Krieger, Gloot and Robles, 2002c; Ghosh, Robles, Glott 2002d; Wichmann, 2002a; Wichmann, 2002b; Wichmann and Spiller, 2002). The reports looked into the motivation of developers, but also the economics of OSS.

4.5.1. Motivations of the developers

Raymond stated:

> Every good work of software starts by scratching a developer’s personal itch (Raymond, 2001, p. 23).

He identified this as being one the most important motivations for OSS developers. However, when other researchers began to investigate the developers’ motivations, they found that status among their peers and the future prospect of a well-paid job were also important (Lerner and Tirole, 2002), while others found that learning and developing skills were important (Ghosh et al., 2002a). In addition, the FLOSS study indicated that there was a difference between the motivations when they joined, when the individual skill and exchange of information was most important, and when commercial and political aspects became more important when the community matured (Ghosh et al., 2002a).

Bonaccorsi and Rossi (2003a) discussed four other aspects of developer motivation. First, they compared creating OSS with academic research, where the result is intellectual gratification, with peer reviews, which was confirmed by Ghosh et al. (2002a). Second, many developers also regard programming as an art form, where the individual contributions have an aesthetic value. Third, OSS developers rediscover the pleasure of creativity, something that has been lost when developing proprietary software. Fourth, the reasoning by Lerner and Tirole (2002) of gaining prestige and visibility is also an aspect for motivation.

However, the motivation of a developer is individual, as Goldman and Gabriel contend (2005). They list ten different reasons why developers volunteer their skills in an OSS environment, ranging from the need for a product (an “itch to scratch” (Raymond, 2001)) to feedback connected to the training of a skill (Goldman and Gabriel, 2005). On the other hand, Goldman and Gabriel seem to miss the fact that an individual may be motivated by several of these reasons, something also indicated by the FLOSS reports (Ghosh et al., 2002a).

Weber lists a number of characteristics needed to motivate developers, based on the product and the contributors themselves. The product and the information about it must be transparent so it is possible to judge the quality and viability now and in the future. The contributors should settle for future rewards, other than economic gain, such as knowledge. In addition, the contributors should be positive, normative, and ethical about the process (2004).

Another interesting aspect having relevance for companies is that in order to motivate developers to contribute to an OSS project, the firm not only needs to release workable code initially (Raymond, 2001), but also commit resources to the project over time (Goldman and Gabriel, 2005). In addition, the developer that initiates the project provides the most crucial contributions over the whole life cycle of the project (Ghosh et al., 2002b).

Dahlander and Wallin (2006) analyzed how firms could unlock complementary assets within OSS projects based on community activities in four OSS firms. Based on earlier research on how some firms succeed with innovations, the authors posited five (four of them major) hypotheses on how firm sponsorship, i.e. employees paid to develop OSS, affects individuals in the OSS community. The result differs from that of other research (e.g. Ghosh et al., 2002a; Hippel and von Krogh (2003) depicted community contributions as essentially altruistic and collaborative, and Lerner and Tirole’s (2002) concept of career boosting. In addition, Dahlander and Wallin found that the network effect was higher for companies with a business model aligned for OSS, opposed to firms with non-aligned business models. Even if a firm
gains the ability to affect the community, communities also have safeguards aimed against (visible) firm control, making the proposition a balancing act.

Thus far, for a company to work with the community and OSS, the firm may have been induced by the idea of user innovation and realize the need to investigate the motivation of the developers. A question may remain: “How do we do this?”

4.5.2. User innovation

One of the most potent research streams regarding OSS is the hypothesis regarding user innovation. Researchers had long realized that extensive in-house research and development was a massive barrier to entry into many markets, and very few corporations had the resources to be able to compete. Suddenly, upstarts who got their ideas to the market in a different way beset these giants. Chesbrough (2003) calls this “open innovation,” while Weber considers that OSS changed the world (2004). User innovation research also includes studies regarding the OSS community as a sub-category.

One natural context for open innovation is OSS. The study of community innovation sprouted a number of articles. One of the hypotheses, the private-collective innovation model, where freely shared information was used for the common good, was presented as opposed to invention created from private investment (von Hippel and von Krogh, 2003). Similarly, the researchers studied how users joined and specialized in community projects (von Krogh, Spaeth and Lakhani, 2003). Some also investigated how users profited by freely revealing their innovations (Harhoff, Henkel and von Hippel, 2003), while others looked into how old and new ventures benefited from revealing their source code and thus gaining advantages and benefits from open innovation (Henkel, 2003; Gruber and Henkel, 2004).

Weber, a political scientist, discussed OSS as a process, where innovation and success for a project occur when certain characteristics are present, such as contributions from free sources, a critical mass of users, peer-review, positive network effects, free leadership and an interactive and developing community (2004).

West and Gallagher (2006) found that OSS offered two new structural approaches to open innovation, pooled R&D and spinout, where the former means a commercial company co-develops a product with an OSS community, and the latter happens when the internal intellectual property is no longer considered valuable and is released as OSS. Value is still created for the company, in the first example through better products at less cost, and in the second example through complementary sales through hardware or services.

However, the majority of OSS projects are not innovative; most seem to mimic other projects (Klincewicz, 2005). In addition, OSS commoditizes software very quickly, and there are indications that the commoditization pressure from OSS would stimulate innovation in proprietary software rather than OSS (Murdock, 2006).

4.5.2.1. OSS communities

OSS communities are the people involved in making and using OSS. Perr, Appleyard and Sullivan (2010) define an OSS community as:

[...]The network or ‘ecosystem’ of participants in the creation of the technology and its accompanying intellectual property (IP) (p. 434).

However, even if there is a loose association between different OSS communities, the aggregate is often termed the “OSS community”, which denotes the connection to software with certain licenses that are developed collectively. Even if this thesis does not address the innovation research stream, the impact of the community in commercial OSS companies’ business model is of interest (von Krogh et al., 2003; Dahlander and Magnusson, 2005; Perr et al., 2010).
The question of whether OSS is innovative matters little for individual developers; what does matter for developers are the reasons for them to enter the community and develop software. Even if it might appear that motivations of developers have little to do with the commercialization of OSS, nothing could be further from the truth. In reality, the developers and users are the community, so companies interested in utilizing OSS should be interested to learn how they could motivate developers to work in their projects.

**Community interaction**

Dahlander and Magnusson (2005; 2008) investigated four OSS companies, three Swedish and one Finnish, to study how they interacted with OSS communities. In the first study, they created three different approaches towards the community, *symbiotic*, *commensalistic*, and *parasitic*. The third approach is one where the company is seeking its own benefits only, neglecting the needs or interests of the community, which ultimately could harm the community. On the other hand, the first approach is a win-win for both parties. In the symbiotic approach, the company and the community co-develop the software. In this approach, the company must gain status through active and direct involvement in the community in order to gain legitimacy to influence the development. If the company gains a preferred position, the benefit is obvious:

One way of viewing this is to consider the community as an extended part of the knowledge base of the OSS firm, however outside of its formal span of control (Dahlander and Magnusson, 2005 p. 488).

The second or commensalistic approach is the middle path between the two extremes. A company approaching the OSS community in this way seeks useful input without hurting the community, and thus gains knowledge or is able to use software in the firms’ products.

All three of these approaches entail different managerial challenges, but also provide the firm with different means to influence the direction of the community. Needless to say, the symbiotic approach gives the firm more tools to affect the community, while the commensalistic has fewer opportunities and the parasitic approach offers none (Dahlander and Magnusson, 2005).

In their more recent research (2008), Dahlander and Magnusson expanded their work by developing three themes on how firms relate to the external knowledge creation provided by the communities. These themes, *accessing*, *aligning*, and *assimilating*, directly connect the business model of the firms and how they utilize their communities.

The first theme concerns how firms access communities to extend their resource base, while the second addresses how companies align their strategies with the community. The final theme looks at how firms assimilate work developed in the community to integrate it with a product and share the result (Dahlander and Magnusson, 2008).

It was found that the studied firms approached the theme for community access in two ways, *establishing a community* and *identifying and assessing existing communities*. Similarly, there were two approaches that concerned aligning the strategy with the community, *clarifying ownership* and *influence the direction of the development*. These approaches are connected, since establishing a community often requires existing software to be effective, making ownership important. By making use of existing OSS, ownership is not as important as trying to influence the development in a desired direction. Finally, the authors also found two different approaches to assimilating the result of the community. First, by *feeding back non-strategic source code to the community*, the desired outcome is to gain status and legitimacy. This may increase chances of influencing the direction of the development, but may also spur activity in the community. In the second approach, firms evaluate and select source code from communities, thus “separating the men from the boys”. Both approaches have benefits as well as disadvantages, but a firm may very well utilize both simultaneously (Ibid.).
The authors conclude that using OSS communities is a way for firms to increase resources for innovations, but that the benefit comes at various costs, such as control and influence. In addition, these two studies are of special interest to the author of this thesis, since they share two case companies and because the author, while working for one of them, contributed to the research.

“Opensourcing”

Ågerfalk and Fitzgerald (2008) made an interesting study on how companies utilize OSS communities, a phenomenon they term “opensourcing”. By applying the psychological contract theory (PCT), which has been used for outsourcing, the authors identified three aspects of importance for opensourcing: mutuality and reciprocity of obligations, distinction between psychological and legal contracts, and the focus on individual beliefs and expectations. All three of these are deemed central to the OSS concept, since OSS is co-developed and made without legal contracts (aside from licenses) and personal motivation drives the individual to develop OSS. However, since the second aspect of personal motivation is based on Lerner and Tirole (2002), it is partly contradictory to the findings of Dahlander and Wallin (2006). Nevertheless, not every OSS developer is sponsored, so the personal motivation concept may be valid for many hobbyist developers, even if the authors point out that some developers are employed to develop OSS.

The PCT also states that both the customer and supplier have obligations, and the authors attempted to apply these to OSS developments. For customers, the model has four obligations: requirements specifications, prompt payment, close project monitoring, and project ownership and management support (Ågerfalk and Fitzgerald, 2008). However, translating these obligations into OSS development is not easy, for example the latter obligation of project ownership and management support. Some company-driven OSS projects fulfill this obligation, but the community owns the majority of OSS projects collectively. The trend towards more company-driven projects will no doubt make this obligation more valid over time.

The second obligation, of prompt payment, is also hard to apply to OSS. However, the authors point to employed developers (see also Dahlander and Wallin, 2006) and to the fact that peer feedback can substitute for payment.

Moreover, the traditional OSS development philosophy, “Every good work of software starts by scratching a developer’s personal itch” (Raymond, 2001, p. 23), would make the first obligation hard to fulfill. However, as the authors note, the traditional truth does not necessarily hold true anymore, since corporations that use particular software try to influence projects in a desired direction (see also Dahlander and Magnusson, 2005; 2008).

However, the third obligation of close project monitoring is feasible. Given that the majority of OSS projects are driven through open online resources, such as Sourceforge, interested parties can see extremely detailed project. The authors also point out the changes in OSS development (see also Fitzgerald, 2006, as described below), which indicates that the process is becoming increasingly formalized, particularly in larger projects (Ågerfalk and Fitzgerald, 2008).

On the other hand, the supplier obligations were easier to translate directly to the OSS community. The five obligations are: clear authority structures, taking charge and solving problems, effective human capital management, build effective interorganizational teams, and effective knowledge transfer (Ågerfalk and Fitzgerald, 2008). Although initially, the first obligation may seem contradictory, most (active) OSS projects have a clear authority structure with a “benevolent dictator” at the top (Raymond, 2001, pp. 101-103; see also Bonaccorsi and

---

27 Sourceforge is a web service that provides for OSS development. See sourceforge.net for more information.
Rossi, 2003a). The second obligation is also satisfied; despite the increasingly formalized structure mentioned above, individuals within the project still retain their independence.

The third obligation has been interpreted as the quality and the longevity of the developer in an OSS project. However, the authors revert back to the conception of the OSS developer as a hobbyist, who wishes to gain better skills (as suggested by Lerner and Tirole, 2002; Ghosh et al., 2002a) and not the employed (sponsored) OSS developer (as suggested by Dahlander and Wallin, 2006).

The fourth obligation, to build interorganizational teams, is unparalleled in OSS. Larger OSS projects are renowned for bridging borders between single companies and industries as a whole (e.g. Kroah-Hartman, Corbet, McPherson, 2009). However, the authors focus on the customer-related side, which is another view, albeit not as clear as the example given above.

The final obligation, of knowledge transfer, is also a strong point for OSS. Since many developers are also users of the software, the need never arises, but since the source code is available, often together with documentation and other support apparatus, a knowledgeable new developer has an abundance of information available online.

To test the concept, the author studied three companies that initiated an OSS project, and the obligations were tested through qualitative methods. The findings were then tested quantitatively through a survey questionnaire (Ågerfalk and Fitzgerald, 2008).

The findings showed fulfilling some obligations were associated with open sourcing success, but that there existed a tension between the customer and community that needed to be addressed and managed:

Overall, the customer must achieve that delicate equilibrium between value-creation in terms of successful business model for itself while not transgressing the OSS community values (Ågerfalk and Fitzgerald, 2008, p. 402).

Moreover, there is one important reason for companies to invest in and initiate OSS projects, namely the opportunity to attract and employ top developers from the community.

### 4.5.3. Economics and business of OSS

The economics and business aspects of OSS consist of several sub-categories. First, generic economic theories that apply to OSS business; second, the issue of OSS licenses, which are critical for firms running business based on OSS; and third, the strategic and business models describing how OSS companies conduct their business.

#### 4.5.3.1. OSS economics

This quote effectively sums up one view on the economics of OSS:

> Yet to an economist, the behavior of individual programmers and commercial companies engaged in open source processes is startling (Lerner and Tirole, 2002 p. 198).

At the end of the last millennium, there was much confusion regarding the Internet and its impact on the economy. OSS was closely related to the Internet boom, given Netscape’s decision to release the source code to the community 1998 (Hecker, 1999; Raymond, 2001), but perhaps even more due to the Red Hat initial public offering (IPO) in 1999 (Krishnamurthy, 2005).

Lerner and Tirole (2001) discussed some aspects of OSS that some observers considered a challenge to economists’ existing paradigms, namely: individual incentives, corporate strategies, organizational behavior and the innovative process of OSS. The authors felt that the OSS movement was well accounted for, in particular individual community participation, project management and corporate participation. All of these issues are covered in Lerner and Tirole (2002); however, some aspects highlighted as interesting for future research were opening proprietary code, legal aspects, and sociological aspects.
The FLOSS report also studied corporate participation by observing the world’s 25 largest software companies and their involvement in OSS projects through websites and search engines. Most companies used OSS, but did not contribute. Those that did contribute had different reasons, such as standardization and compatibility issues, making low-cost components in product bundles, and promoting non-essential software to OSS (Wichmann, 2002b).

In a later work (2005a), Lerner and Tirole summed up the older aspects, addressed two of the earlier highlighted ones, and added a few new ones. The first aspect concerned opening proprietary code, termed code release strategy, is where the network releases former proprietary software as open. The strategy is similar to the razor business model, where the razor body is given away or sold at a substantial discount, whereas the necessary razor blades cost more.28 The idea is that the development in these projects can be used to bootstrap services, hardware or proprietary software products. The strategy is also relevant if the software is lagging behind the market leader, e.g. Netscape’s decision to open its web browser product (see Raymond, 1998; 2001). In addition, they summarized their other article on OSS licensing (Lerner and Tirole 2005b). This article is further discussed below.

Another issue discussed concerned the relative quality and security between OSS and proprietary software (Lerner and Tirole, 2005a). Some efforts have been made to substantiate Raymond’s (2001) claim that OSS has higher quality and security than proprietary software, based on “Linus’ Law”, but no definitive answers have been found.

Another area where there were no definitive answers concerned governmental policies regarding OSS. Several governments have sought to encourage OSS in various forms; the question is whether this is beneficial (e.g. Wichmann, 2002b). Some researchers indicate that there are positive effects, while others show opposite results – there is no definitive answer (Lerner and Tirole, 2005a).

Another aspect that was important in 2005 was the ongoing litigation from SCO and prominent users of Linux. The issue of software patents and the impact on OSS was sparsely studied, and some worried about the recoil on OSS development. At this time, some larger OSS vendors initiated software assurance programs aimed at protecting their customers. In addition, patent holders started to pool patents connected with OSS in order to offer further protection (Ibid). Even if the SCO litigation was finished (Paul, 2010), the sale of Novell still left the question unanswered (MacDonald, 2011).

The article also discussed open source in a wider context, i.e. if the concept could be translated into other industries, and how companies might benefit from the concept in other ways. Here, Lerner and Tirole give three examples for how a company could benefit by adopting similar arrangements to those of OSS. First, open-source technology enables users to tailor the technology to their specific needs without suffering from future price increases due to lock-in effects. Second, the supplier may avoid patent litigation from overlapping IPR. Third, by releasing a technology as open source it may help standardization. However, other methods exist today to avoid these problems, which are costly for smaller firms; it is here that utilizing the open-source concept can be a low-cost alternative. Finally, the authors also addressed academia as an area where some arrangements are similar to those of open source, while others are more proprietary, particularly in the economics community (2005a).

Lerner and Tirole have contributed greatly to the collective knowledge about OSS, securely based in traditional economic theories. However, some aspects of OSS are not as well covered, for example how actual companies make business out of OSS. Nevertheless, as other re-

---

28 The same principle has been used for inkjet writers, where the inkjet machine is sold cheap, while the ink is expensive.
searchers cover these aspects, Lerner and Tirole’s efforts will boost the OSS research community for years to come.

**Statistical testing**

A number of articles have presented attempts to utilize various formulas to explain different phenomena in OSS. One particular subject is to determine the feasibility for firms to use or adapt OSS in different scenarios such as software development (Johnson, 2001; Harison and Cowan, 2004), comparisons between OSS and proprietary software (Dalle and Jullien, 2001; Mustonen, 2003; Bonaccorsi and Rossi, 2003b), the effects of OSS for a competitive firm (Hawkins, 2004), OSS adaption (Chi, 2007), and dual licensing (Comino and Manenti (2007). These articles usually show, with differing degrees of clarity, that OSS is very well feasible in these settings, and that OSS and proprietary software will likely coexist, rather than subsuming the other (*e.g.* Bonaccorsi and Rossi, 2003b). However, the scenarios are of course simplified, which hampered their usability for this thesis. Nevertheless, these articles span a theoretical bridge that enables an understanding of the underlying theorems that make OSS “tick.”

### 4.5.3.2. OSS licensing

OSS licenses play a central role for commercial OSS companies. When Richard Stallman created the GPL, he made use of the Copyright Laws of the United States. His agenda was to ensure that the source code would be accessible, and that no one would be able to hoard free software in order to make it proprietary and make a profit without benefiting the original authors (Stallman, 2001).

His creative use of the law and the characteristics of the license have caused different reactions from various agents. Microsoft condemned it and Linux as un-American, communistic or a cancer (Greene, 2001; Lettice, 2002).

The “infectious” part of GPL has been discussed in and outside of the OSS community. Vetter (2004) shows that the “hard” terms may hurt the OSS movement, since the balance between OSS and proprietary software is impaired, pointing for example to Linus Torvalds’ modification of GPL 2.0. He considers the newer OSS licenses as better balanced for this matter.²⁹

While Raymond and Stallman argue that the “Direct-Sale Value Model” is not applicable for FS or OSS, the fact remains that the copyright owner of software may retain the right to use a proprietary license alongside GPL (Stallman, 2001; Raymond, 2002; Bonaccorsi and Rossi, 2003c; Välimäki, 2003).

Välimäki (2003) describes three cases with dual licenses, i.e. the same software has two licenses: one OSS license, and one proprietary license aimed at companies wishing to embed OSS inside their proprietary software. By granting these firms a proprietary license, the viral effect of GPL, for example, is nullified. At the same time, the companies using this scheme often use the GPL license. The use of GPL is believed to have a strong network effect for the companies involved, which includes marketing, political and developmental benefits. However, the legal requirement for dual licensing is that the “company has undisputed rights to the software product it wishes to dual license” (Välimäki, 2003 p. 71).

In an Italian study, the local software industry was researched with respect to OSS, and in particular, how firms entered the industry and the licensing schemes they used. The researchers noticed that firms actively participating and working with OSS invested more in OSS but

---

²⁹ Even if it may appear so, Vetter does not miss Stallman’s points with GPL, but he argues for the benefit of the OSS initiative rather than FS, so his article could be seen as a part of the discussion between OSS and FS.
more frequently used copyleft licenses. They also observed that very few firms had adopted a pure copyleft licensing revenue model. About 60% of the firms combined the copyleft license with other licenses, including proprietary (Bonaccorsi and Rossi, 2003c; Bonaccorsi, Giannangeli and Rossi, 2006).

The abundance of copyleft licenses was attributed to the use of Linux-based products, strength of social motivations, and various engagements in the OSS community. Most of the firms had active relationships with the community. While the firms did not contribute much to the production process of OSS, they were much more involved in creating a larger user community, thus adding to the distribution process of OSS. The authors also noticed that since the firms used mixed licensing models, the implied worry about the “viral” property of the copyleft licenses was not justified. The studied firms were instead well aware of the legality of different licensing schemes (Bonaccorsi and Rossi, 2003c).

Välimäki and Oksanen (2005) studied the three main operating systems for desktops, Microsoft Windows, Apple OS X and GNU/Linux. They also studied the differences in these companies’ business models. They concluded that the market is still dominated by Microsoft, which profits from its proprietary license and development model. However, Apple OS X, using a mixed license (based on FreeBSD) and development model and GNU/Linux has affected the market, creating open standards and stability for the operating systems. Even if the open-source development model is more socially acceptable, the proprietary model still generates more revenues.

Lerner and Tirole (2005b) studied the scope of OSS licensing and found that, in opposition to proprietary software, the decision on which license to use was not just based on the preferences of the licensor itself, but also had to consider the community of users.

For instance, a commercial company releasing software to the open source community may choose a more restrictive license because of suspicion about its ultimate intentions (Lerner and Tirole, 2005b, p. 53).

They also found that software geared towards software developers more often had less restrictive licenses, but also that less restrictive licenses attracted more developers than those more restrictive (Ibid).

However, since the sample was based on activities on Sourceforge, some of the oldest and most active OSS projects were not included in the study, for example the Linux Kernel and Apache projects. These projects attract many developers and companies, despite using different kinds of licenses. Nevertheless, Sourceforge is the largest web-based source code repository for OSS development, and the study gives interesting insights regarding the relationship between choice of license, activity and target audience. This research does not only have relevance for companies considering “going open”, but also innovation and motivation for individual developers (Ibid).

Goldman and Gabriel (2005) take a wider stance on licensing and discuss not only licenses for OSS, but also semi-open licenses, called gated communities, such as Microsoft Shared Source Initiative and the Sun Community Source License (SCSL). In addition, they also put forward comparisons between different OSS licenses, and discuss if dual or hybrid licensing is possible.

Fitzgerald (2006) points out a shift in OSS licensing in his article. He predicts that corporate-type licenses, for example Mozilla Public License (MPL) and IBM Public License (IPL),

---

30 Examples of copyleft licenses are General Public License (GPL), Library (or lesser) General Public License (LGPL), and Mozilla Public License (MPL).

31 The SCSL was originally used for technologies such as Java, but it was released under a new, more open license in 2006 (cf. LaMonica, 2006a).
are becoming more important. He also foresees that border licenses, such as the Microsoft Shared Source Initiative, will be more common. However, the development of GPL Version 3 and the initiative from Sun Microsystems indicates that even large companies are more likely to accept the reciprocal attributes GPL offers and the signals it sends to the OSS community (LaMonica, 2006a).

Olson (2006) discusses the dual licensing strategy as being one of three for commercializing OSS. The other two are service business, where the company offers service to OSS, and hybrid business, where the company uses the OSS as a platform and adds proprietary software from which the revenue is collected.

Olson considers dual licensing as an innovative way to combine the benefits of both OSS and proprietary software. This can be done by using the availability of the OSS to create a large, inexpensive distribution network while still gaining revenues for the software, as well as enable the bundling of third-party software to the product. He points out that the company using dual licensing has to own the copyrights to their own products, which makes it harder for them to make use of community contributions. He states, “dual-licensing business relies on open source as a distribution strategy, not as a production strategy” (2006 p. 79).

The last statement may be somewhat moderated from case to case, since firms can utilize other strategies, such as different OSS licenses, which enables them to make use of the community contributions.

4.5.3.3. Enforcing the copyleft

What happens if someone breaks the license agreement of OSS? Does the copyright owner have any protection? Proprietary software developers usually use legal means to sue the perpetrator, but the practice differs for OSS. Their approach is more moderate, given their aim to force the offending company to release the source code for the benefit for the whole community (Välimäki and Oksanen, 2006). The stricter enforcement of copyrights in the developing countries (which show very high rates of pirated copies of proprietary software) may bolster the development of OSS in these areas. There are already examples of developing countries actively promoting OSS in order to boost their software development (Ibid).

In effect, the copyleft licenses could be considered as promoting competition. Välimäki studied how copyleft works in light of the EC Competition Law, and found it was closer to the goal of the laws than the current proprietary licenses (as exemplified by Microsoft licenses) (Välimäki, 2006).

4.5.3.4. OSS strategies and business models

As discussed in the previous chapter, this thesis does not consider strategies and business models as disparate, but rather complementary. Strategy denotes certain actions to attract customers, stave off competitors or similar direct actions that concern interacting with entities outside of the firm, while business models describe how the firm delivers value to their stakeholders, including ways to produce offerings and gain revenues. With this said, strategies are not excluded from the business models, but are considered to be a part of the value creation of the firm.

OSS strategies

Raymond (2001) outlined some strategies for OSS companies, but his efforts were aimed more towards business models. Schiff (2002) summarized the early efforts and pointed out that even if Raymond (2001) made a list of existing or possible business models, Lerner and Tirole (2002) provided a more heuristic approach and presented three strategies viable for commercial OSS companies to approach OSS: living symbiotically off an open-source project, code release, and intermediaries. However, these strategies are described in such a
way that they can be translated into what other authors call business models. The descriptions presented below aim to do just that.

The first strategy, which is to provide services for an existing OSS project, is akin to what Raymond considers giving away the recipe (2001), and is also a strategy that Stallman advocates (2002). Others share similar concepts (Hecker, 1999; Wichmann and Spiller 2002; West, 2003; Goldman and Gabriel, 2005; Krishnamurthy, 2005; Perens, 2005).

The second strategy of code release, i.e. when a company releases proprietary software in order to promote sales in a related area, is also suggested by Bonaccorsi and Rossi (2003c) and Dahlander (2005) as a part of “sharing competences” with the community. The strategy is similar to “widget frosting”32 (Raymond, 2001; Hecker, 1999), and may be seen as a bundling of products (Wichmann, and Spiller, 2002; Goldman and Gabriel, 2005), while West (2003) considers a strategy where a firm partly opens software in order to promote proprietary products (a version of shareware).

The last strategy Lerner and Tirole suggest is intermediates (2002). The concept allows a firm to sell services, based on information or other sources, like a broker (Hecker, 1999; Goldman and Gabriel, 2005, Krishnamurthy, 2005).

One important part of business models and strategies for OSS addresses how to balance what code is open and what is not. The protection of IP has been one of the toughest nuts to crack regarding commercial interests and OSS. West (2003) investigated three multinational corporations and their strategies regarding some of their software. Apple, Sun Microsystems, and IBM adopted different strategies for their proprietary and OSS platforms. Where IBM adopted Linux through its entire hardware platforms in its shift to become more service-oriented, and where software commoditization would not hurt its revenue streams, both Apple and Sun had to resort to different strategies. Interestingly enough, these companies adopted two different hybrid strategies, as West refers to them.

Apple decided to base its new operating system (Mac OS X) on the OSS operating system Free BSD; the company also decided to grant all rights for a subset of its new system. West calls this strategy “opening parts”. Sun chose another direction; it released its entire source code for Java (a programming platform) and Solaris (an operating system), but used rather restrictive licensing terms. West refers to this strategy as “partly open”. Sun also “submitted” to the Linux hype and started to offer Linux on their hardware, in a strategy called “if you can’t beat them, join them” (West, 2003).

West also proposes a “three-stage evolution” for proprietary vendors to go to OSS. The time during which a company can retain proprietary software is limited and only viable for a few market leaders. A starting point is to modify the software to incorporate open standards. The next step is to adopt one of the hybrid standards before becoming fully OSS (West, 2003).

Even if West’s article is some years old, the strategies for the chosen companies have become even more polarized. IBM still considers OSS as a key strategy, one which opens up new technologies and platforms (IBM, 2008). Sun, for instance, changed strategies and opened up all of its technologies to OSS, even adapting GPL as its license of choice, and purchased MySQL (LaMonica, 2006a; MySQL, 2008; O’Brien, 2008). Apple, on the other hand, seems to have changed back to a more proprietary strategy, perhaps fueled by its successes from the iPod and iTunes (Burrows, 2006).

Goldman and Gabriel (2005) argue that companies owning the copyright of the source code and use dual licensing or other hybrid-licensing models should expect less in the way of

---

32 Raymond uses the term ”widget frosting” for when companies release the drivers to their hardware as OSS (2001).
community contributions. This may very well be the case, but in some circumstances, it may be difficult to gauge. There are companies that have commercial models and yet retain a large community of developers; perhaps the developers have begun to accept the fact that OSS is becoming more commercial? Perr et al. (2010) conclude that an OSS vendor is less inclined to alienate its community, since it could impair future technical superiority.

Campbell-Kelly and Garcia-Swartz (2009) studied IBM’s software strategies from the 1950s to the present day and found that the computer giant’s strategy had changed significantly during that time due to greater openness. The authors concluded that the last swing towards OSS was based on pragmatism, not ideology, and due to four discerning aspects. First, supporting Linux meant that IBM adapted to the new era of client-server computing, which emerged at the end of the millennium. Second, by adapting Linux, IBM worked towards a unified platform for all its hardware platforms. Third, by adapting Linux (and other OSS) IBM could boost the demand for IBM middleware. Fourth, the shift towards OSS meant an alignment towards services. All of these aspects were geared towards boosting revenues, but in addition to this, IBM was also able to reduce cost. This was basically accomplished through OSS development and also by streamlining the internal operating system platforms.

OSS business models

Armed with the concepts of business models proposed by Raymond (2001) and Stallman (2002), researchers studied existing companies and attempted to evaluate existing strategies and suggest new ones. Numerous strategies and business models have been suggested; however, most of them have been influenced by the suggestions made by Raymond (2001). These are summarized in Fel! Hittar inte referenskälla.: Raymond has influenced a majority of the researchers, but in addition, different authors have presented a number of other concepts.

<table>
<thead>
<tr>
<th>Use-Value Funding Models</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Cost-Sharing</td>
</tr>
<tr>
<td>Risk-Spreading</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Indirect Sale-Value Models</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Loss-Leader/ Market Positioner</td>
</tr>
<tr>
<td>Widget Frosting</td>
</tr>
<tr>
<td>Give Away the Recipe, Open a Restaurant</td>
</tr>
<tr>
<td>Accessorizing</td>
</tr>
<tr>
<td>Free the Future, Sell the Present</td>
</tr>
<tr>
<td>Free the Software, Sell the Brand</td>
</tr>
<tr>
<td>Free the Software, Sell the Content</td>
</tr>
</tbody>
</table>
Hecker, one of the systems engineers that released the Netscape source code in 1998, adds two concepts he calls “hybrid business models” (1999). The first concept has been dubbed “dual licensing,” since the same software has two different licenses. Here, customers may purchase a proprietary license in order to use the software in their own proprietary product, while normal users get an open license (Wichmann, and Spiller 2002; Dahlander, 2005; Goldman and Gabriel, 2005; Krishnamurthy, 2005). It can also be used as a way of mixing open and closed source software (Perens, 2005). The second hybrid concept is to allow free use of the software, but to restrict or charge for modifications of the code or when distributing the modified software (Hecker, 1999). Essentially, this is a variation of the freeware concept, but with the source code included. However, considering how hard this would be to control, the feasibility for such a model is diminished. In addition, the concept of OSS, where contributions are an essential part of the idea, would be lost using it. All of these ideas have been criticized, since it reduces the freedom of the user by the use of closed software (Stallman, 2002).

Most of the business models described above have been based on theory. The FLOSS study investigated 25 firms and their involvement with OSS (Wichmann, 2002b). Based on the data, Wichmann and Spiller present a number of “best practices” (2002). The business models are comparable to those of the other authors, but the companies have been examined based on their offerings and the revenue models. The result was to divide the companies into distributors (retailers) and service providers and map their different business models accordingly (2002).

Aside from discussing nine open-source business models, based on earlier writings (mainly Raymond (2001) and Hecker (1999), Goldman and Gabriel (2005) try to take a holistic view on how a company could approach OSS by dismantling myths about OSS. They list four reasons why a firm should consider OSS as a viable option for software development: improved communication with customers, marketing, external help with support and development, and improved innovation and quality. In addition, they propose another business model, the subscription, where the firm provides a service which updates the customer’s systems with tested software (2005).

Krishnamurthy (2005) presents three basic business models for OSS with variants to combine products and services to gain revenue streams. For the distributor model, he envisions three service alternatives: retail, enterprise support, and upgrade. For the producer model, there are two different approaches: the non-GPL model, where the OSS is embedded in proprietary product, and the GPL model, where the company still uses proprietary licensing, but also provides the source code for their users. The main difference “is in terms of what the seller expects from the user” (Krishnamurthy, 2005, p 284). For the final basic model, the third-party service provider, Krishnamurthy presents two different strategies: selling software combined with services or only offering services. The first strategy includes part OSS and part proprietary software (Raymond, 2001; Wichmann and Spiller, 2002, etc.), while the latter is straightforward services (2005).

One of the pivotal figures of OSS is Bruce Perens, who together with Eric Raymond founded the Open Source Initiative (OSI). His policy document for the Debian Linux project became the Open Source Definition, which defines what software licenses are acceptable for OSS. Perens investigates the role of software in business and the paradigms for software development and their effectiveness. He demonstrates that the retail paradigm, the one connected to Microsoft, is really a small part of the total production of software. He also describes the ineffectiveness of that paradigm, and argues in which cases the OSS paradigm is more effective. His view is that software that does not differentiate a business is more cost-effective than all other paradigms. Moreover, since most software is of this category, most companies would benefit by using and contributing to OSS (Perens, 2005).
Perens then discusses who contributes to OSS and how commercially effective the different contributors are and identifies a number of different business strategies:

- **Proprietary open-source** – the company is a software integrator or distributor that packages its services in a similar way to proprietary software vendors.
- **Mixed open-source and proprietary software** – the company offers the same software under different license models [termed dual licensing below].
- **Open-source with proprietary accessories** – the company adds proprietary components to an OSS.
- **Open-source plus services** – the company offers services for a particular OSS.
- **Hardware vendors** – the company manufactures hardware and subsidizes OSS development.
- **End-user business plus contractors** – the company makes use of OSS in order to provide online content.
- **Service business** – the company customizes OSS packages for a particular customer or service for a selected collection of OSS.

Even if Perens’ main discussion is about how different entities contribute to OSS development, his descriptions of the commercial interests of OSS provide us with insight into these matters (Perens, 2005).

Dahlander (2005) compares the commercialization of OSS with the commercialization of academia and the commons. Even if the focus of the article is on the comparison between academia and OSS, while studying new entrants within the OSS market he found three strategies for how the companies commercialize OSS:

- **Founding projects** – the company releases source code under an OSS license in order to gain resources for developing and testing.
- **Sponsoring projects** – the company sponsors one or more OSS projects that are relevant for its business. The sponsorship comes in different forms, ranging from computer equipment to paid employees.
- **Using projects** – the company makes use of existing code in its own products or offerings.

Since Dahlander’s focus has been on the tension between commercialization and the commons, he has not discussed the strategies in depth, but the scope of his investigation makes it interesting to see how new entrants relate to OSS.

There is no doubt that the landscape of OSS has changed since commercial interests staked claims in the community. Fitzgerald (2006) has dubbed the term OSS 2.0 in order to describe the changes. He presents the evolution throughout the whole concept of OSS, from development to business strategies and licensing.

From the point of view of this research, Fitzgerald’s analysis is spot on when he describes the changes in not only the way the software is produced, but also in changes regarding the attitudes of actual users of OSS customers, who are now willing to pay for professional support. In addition, he presents the business strategies that he feels have evolved from earlier strategies. The strategies are grouped into four categories: value-added service enabling, market creating, leveraging community development and leveraging the open-source brand.

Value-added service enabling treats OSS as a platform and enables the company to bootstrap its special services or competence on top of the OSS.

There are four different market-creating strategies. The first is loss leader, where a company subsidizes OSS in order to create a market. The second model, dual licensing, means the

---

33 In this case, "bootstrap" connotes the possibility to add abilities a firm normally would not be able to without OSS.
company offers two different licensing options for the same software. The third is cost reduction, where OSS provides low cost, reliability and interoperability between platforms, thus reducing costs for a customer. The fourth is accessorizing, where firms offer products other than software, for example books (Fitzgerald, 2006).

A company can leverage community software development and thus receive free programming, which increases the functionality of the product. Fitzgerald also calls this phenomenon circular, since the greater the functionality of the software, the more interest it attracts, thus completing the circle.

The final strategy, leveraging the open-source brand, stems from the fact that government institutions increasingly prioritize OSS in their solutions. It is therefore important for a company that focuses on government customers to align its brand to OSS (Ibid.).

Another important economic factor for a commercial OSS company is how to handle its IPR. In addition, the revenues are directly dependent on which OSS license the firm uses, since the license may narrow or widen the possibilities for different models to gain revenue. Therefore, OSS licensing plays an important role in the firm’s commercial success.

Bonaccorsi et al. (2006) studied hybrid business models in the OSS industry, and in particular entry strategies. Business models are categorized into “pure” or “hybrid”. A pure business model is either proprietary, gaining revenues through software licenses, while a pure OSS business model gains revenues from services. A hybrid business model is a mix between pure business models. Based on a survey consisting of 146 Italian firms from 2003, the findings were that a majority of the companies utilized a hybrid business model, and that experience from working with OSS projects influenced the openness towards OSS communities, which had been negatively influenced by switching costs and network externality effects. They also found that the hybrid business model was not a transient model between the pure business models, but rather a permanent feature. These findings have three implications on management decisions regarding working with OSS. First, the decision to adopt OSS is not irrevocable; proprietary and OSS solutions can be evaluated case by case. Second, OSS can help firms to actively promote and support customer involvement in product development, which will increase the perceived value of the product without raising the marginal costs. Third, it is not a consistent strategy to be a “free-rider” since community participation will increase learning and advantages when interacting with customers. The results are consistent with a shift perspective of OSS, something that other researchers have noted (see also Fitzgerald, 2006).

Campbell-Kelly and Garcia-Swartz (2010) proposed three hypotheses stating that OSS vendors’ business models and strategies are converging towards those of proprietary software based on three criteria, production of software, business models, and strategic interactions. First, in terms of production of software, the authors identify how R & D investments and acquisitions lead OSS companies to have similar expenditures as proprietary companies. Second, in terms of business models, the authors discuss how the dual-licensing revenue model has changed the focus from services for OSS vendors as the way to gain revenues, towards proprietary software vendors’ revenue models. Third, the authors study the interaction between OSS and proprietary software vendors, and state the following:

---

34 Business models are defined as “the way products/services are sold to customers, cash is generated, and income is produced” (Bonaccorsi et al. 2006, p. 1086).

35 Here, business models are connected to the offering and revenue generation, as described in Campbell-Kelly and Garcia-Swartz (2010, p. 225).
We argue that OSS companies are slowly penetrating the software stack from the bottom up and, in the process, are establishing relationships of competition and cooperation with proprietary companies (Campbell-Kelly and Garcia-Swartz, 2010, p. 225).

The article compares proprietary software and OSS vendors from the computer stack from the bottom up using major vendors like Microsoft, Oracle, IBM and SAP as examples of proprietary software vendors, while using Red Hat and MySQL as examples of leading OSS vendors. The hypotheses presented by the authors are thought-provoking, and in line with the findings of other authors (for example Bonaccorsi et al., 2006) and the concept of changes in the view of OSS, as described below.

Perr et al. (2010) studied emerging business models for OSS companies. Their definition of business models also went beyond the majority of researchers by stating:

[…] we define ‘business model’ as not merely the method(s) by which companies derive their revenue, but also the accompanying set of business processes and organisational arrangements required to realise the revenue model. Business models allow organisations to capture the economic value associated with their product and service (Peer et al., 2010, p. 434).

Based on three categories, licensing and IP issues, community profile, and target market, the authors identified seven profitable business models: professional services/consulting, support, subscription, dual license, hybrid/proprietary extensions, device, and community source/consortia. The article notes that OSS business models mirror the trends in proprietary software, and that some observers attribute this to OSS itself (cf. Campbell-Kelly and Garcia-Swartz, 2010).

OSS business models can be confusing, since the definition of business models varies. However, all OSS business models consist of OSS community interaction in some form. In addition, Raymond (2001) inspired many early researchers; therefore, Table 3 below has been constructed based on his concepts, while those of others have been mapped. The hope is that this will clarify the similarities and differences in this area.
## Table 3: Business models for OSS.

<table>
<thead>
<tr>
<th>Authors</th>
<th>Use-Value Funding Models</th>
<th>Indirect-Sale Value Models</th>
<th>Direct-Sale Value Models</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Raymond (2001)</strong></td>
<td>Cost Sharing</td>
<td>Give away the recipe</td>
<td>Free the software, sell the brand</td>
</tr>
<tr>
<td><strong>Hecker (1999)</strong></td>
<td>Loss-Leader</td>
<td>Support Seller (Software franchising)</td>
<td>Accessorizing (Software franchising)</td>
</tr>
<tr>
<td><strong>Lerner and Tirole (2002)</strong></td>
<td>Code release</td>
<td>Symbiotic subsidizing</td>
<td>Intermecenates</td>
</tr>
<tr>
<td><strong>Stallman (2002)</strong></td>
<td>Selling media, not the software</td>
<td>Consulting services</td>
<td>Niche and specialty OSS distributors</td>
</tr>
<tr>
<td><strong>Wichmann and Spiller (2002)</strong></td>
<td>Original Linux distributors</td>
<td>OSS distributors and complimentary product</td>
<td>OSS conference organizers</td>
</tr>
<tr>
<td><strong>West (2003)</strong></td>
<td>Partly open</td>
<td>Opening parts</td>
<td></td>
</tr>
<tr>
<td><strong>Dahlander (2005)</strong></td>
<td>Sponsoring projects</td>
<td>Using projects</td>
<td></td>
</tr>
<tr>
<td><strong>Goldman and Gabriel (2005)</strong></td>
<td>Bundle OSS with other software + Hardware</td>
<td>Consulting services</td>
<td>Ancillary items</td>
</tr>
<tr>
<td><strong>Krishnamurthy (2005)</strong></td>
<td>Distributor (1)</td>
<td>Distributor (2/3) + Third-party Service Provider</td>
<td></td>
</tr>
<tr>
<td><strong>Perens (2005)</strong></td>
<td>End-User Business + Contractors + Service Businesses</td>
<td>Open Source with proprietary accessories + Hardware vendors</td>
<td>Open Source + Services</td>
</tr>
<tr>
<td><strong>Bonaccorsi, et al. (2006)</strong></td>
<td>Pure OSS business models</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>West and Gallagher (2006)</strong></td>
<td>Pooled R&amp;D</td>
<td>Selling complements</td>
<td>Spinout</td>
</tr>
<tr>
<td><strong>Perr et al. (2010)</strong></td>
<td>Community source/consortia</td>
<td>Professional services/consulting</td>
<td>Device Hybrid/proprietary extensions Support</td>
</tr>
</tbody>
</table>
To summarize and illustrate how the above-referenced researchers are connected to the three main research streams, Table 4 below shows which work concerns which research stream:

Table 4: Authors and research streams in OSS.

<table>
<thead>
<tr>
<th>Authors</th>
<th>User innovation</th>
<th>Motivation of developers</th>
<th>Economics of OSS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bonaccorsi and Rossi (2003a)</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Bonaccorsi and Rossi (2003b)</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Bonaccorsi and Rossi (2003c)</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Bonaccorsi et al. (2006)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Campbell-Kelly and Garcia-Swartz (2009)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Campbell-Kelly and Garcia-Swartz (2010)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Chesbrough (2003)</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dahlander (2005)</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Dahlander and Magnusson (2005; 2008)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dahlander and Wallin (2006)</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Ghosh et al. (2002a; 2002b)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Goldman and Gabriel (2005)</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Gruber and Henkel (2003)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fitzgerald (2006)</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Harhoff et al. (2003)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hecker (1999)</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Henkel (2003)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Klincewicz (2005)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Krishnamurthy (2005)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lerner and Tirole (2001; 2002)</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Lerner and Tirole (2005a)</td>
<td></td>
<td></td>
<td>(X)</td>
</tr>
<tr>
<td>Lerner and Tirole (2005b)</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Murdock (2006)</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Perens (2005)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Perr et al. (2010)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Raymond (2001)</td>
<td>(X)</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Schiff (2002)</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Stallman (2002)</td>
<td></td>
<td></td>
<td>(X)</td>
</tr>
<tr>
<td>Von Hippel and Von Krogh (2003)</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>West (2003)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>West and Gallagher (2006)</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Wichmann (2002a; 2002b)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wichmann and Spiller (2002)</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Ågerfalk and Fitzgerald (2008)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The research regarding OSS covers different aspects of the phenomena, with insights as well as conflicting views. The material also varies between theoretical tests and assumptions and empirical multi or single cases. A number of the researchers are studying business models, but very few have coupled the business model research to the OSS research. The aim of this the-
sis is to do just that by analyzing real OSS vendors through the use of a business model re-
search framework.
5. OSS business model analysis framework

When comparing the business model concepts presented by OSS scholars with those from academia, the former is lacking. Therefore, a new business model framework would be important, drawing on academic precursors to provide the depth and insights that the OSS models lacked. The framework presented here is an adapted framework based on the model used in an earlier thesis (Rosén, 2008), which in turn was adapted from a colleague (Kindström, 2003). The latter framework was used in the pre-study, below, but some shortcomings were noted during that time. The solution was to create a new framework, one that could be used to observe and analyze OSS companies.

The primary reason for the adaptation was the OSS communities. The communities, which consist of non-paying users and developers who test and develop OSS software gratis, are vital for OSS companies. In fact, the contributions made by certain communities are so important for the firms, that they should be considered as a part of the company’s microenvironment. The impact of the communities is unique for the OSS industry, which alone justifies a new framework. However, the implications of the OSS community may soon spread into other areas, as suggested by some scholars (Weber, 2004).

A second reason was the fact that copyright laws and some OSS licensing limit the number of revenue models, such as direct-sale value models used by proprietary software companies. Therefore, offerings are of importance where revenue and distribution models are different in the OSS industry as compared to other industries. Based on these findings, a business model framework was constructed and used in the licentiate thesis (Rosén, 2008).

Following completion of the licentiate thesis, it was determined that there was a need to further improve the model by adapting it to study OSS vendors. Therefore, clarifications have been made from the previous model, which consist of a stronger focus to ascertain configurations of the business model.

There are still limitations with this framework, for example that it is aimed towards a single business unit or firm. A conglomerate or multidivisional corporation may have different business models for each department, and these may or may not be managed through a corporate strategy. In some cases, even very large corporations may operate using a primary business model. In addition, a firm or a business unit may very well have more than one business model, but there is usually one primary model that may be coupled with other supporting business models.

One aspect of creating a specific, yet customizable analysis framework is to highlight important theories by the researcher. It is the author’s firm belief that one theorist, sadly no longer with us, should have influenced business theory to a greater extent. That theorist was Richard Normann (1943 – 2003), whose concept “affärside” (loosely translated to “business idea”) describes a system for dominance for a firm (Normann, 1975). Several of his concepts can be found in many of the more known management theories created after 1975. More importantly, his theories were thoroughly tested, since he was a celebrated consultant for many corporations. However, some of his later concepts have also received attention, particularly the concept of customer orientation (Normann and Ramirez, 1994; Normann, 2001). All of his concepts will be incorporated in this business model framework.

The business model framework will therefore consist of three different elements: business positions, operational platform, and offering. Figure 4 below illustrates the framework.
5.1. Business positions

The element called “strategic positions” in Kindström (2003; 2005) has been renamed “business positions” for clarity. Its principles are the same; it contains embedded strategic elements such as market, brand and positioning, and customers, partners, and market channels are clearly influenced by the concepts introduced by Porter (1980) and the five forces that affect an industry.

In this framework, the business positions are abstract descriptions of how the firm and the macro environment interact with one another. As such, these positions should never be stable, no matter how they are perceived from the outside.

In the pre-study, most of the companies described how they started from an ideological viewpoint and then moved on from there, which corresponds to the ideological strategy in Mintzberg and Waters (1985). However, they often found themselves unable to affect the strategies imposed by the environment, and found they were just “playing along,” and finding working solutions, something which corresponds to the imposed strategy typology (Mintzberg and Waters, 1985).

5.1.1. Business ideas, visions and goals

In this framework, the business positions covers the concepts of business ideas, visions, and goals and aspired positioning.

In Normann’s (1975) concept of the business idea, a fully functional firm has to have a congruence of the actual market segment, the product offered and the internal structure and control systems of the firm. It can also be seen as a manifestation of how the firm is presently working (Normann, 1975). In this framework, the business idea directly influences the operational platform, through strategies and plans.

Normann (1975) also described the vision of the company; one important aspect is that it is never supposed to be made permanent. On the contrary, it is supposed to change over time. It is not just another set of goals, but rather something to aspire to. The vision can be utilized in three ways in the firm: it controls learning by pointing out the next step of the learning process; its changes are a measurement of the learning; and it facilitates the creation of inter-
nal driving forces in terms of ambition and engagement (Normann, 1975). Just like the business idea, the vision influences the operational platform in this framework, through strategies and plans that mainly concern learning.

The concept of goals should be familiar to most businesses. However, the way in which goals are formulated or measured differs between all firms. The goals are often used as a means to control or encourage behaviors in the firm within a shorter time span than visions. The goals are also set in a variety of ways, but often through benchmarks against a competitor or market leader. However, goals have traditionally been measured through financial perspectives, while other aspects have been downplayed. Several efforts have been made to handle this; one example is the balanced scorecard system (Kaplan and Norton, 1992). However, in this framework, the goals or the systems are part of the strategies and plans that affect the operational platform in the form of written strategies or plans.

The *aspired position* is the firm’s view of itself on the market, when addressing its competitors by, for example, pricing. By setting a price on its offering, the firm asserts a position on the market. This mechanism is called *pricing strategy* in this thesis, and is based on Porter’s generic strategies (1980), which consist of two different strategies, cost leadership and differentiation. Cost leadership concerns a low-price strategy and economies of scale where the company competes through lower prices, while differentiation is when a product or firm is perceived as different or unique from the rest of the industry, and successful differentiation allows for higher prices. Porter also adds that the term “focus” means a firm is targeting a certain market segment to serve it as well as possible, and uses cost leadership or differentiation in this niche. However, if the firm does not follow one strategy, there is an increased risk that it will be “stuck in the middle” (Porter, 1980).

Therefore, companies usually strive to utilize one of the generic strategies, and this affects the business model. For example, if the firm considers itself a “cost leader” this will affect the operational platform by focusing on efficiency, for example lean engineering, and the offering itself, through a low-price pricing model.

This framework makes use of these theories, albeit with some alterations. In this framework, the business idea is the firm’s *view* of its own business – the reason for its existence. Additionally, the use of vision considers the goals and aspirations themselves, rather than its written plans. These written strategies and plans are considered part of the *operational platform* (see below).

In order to capture the intended strategy of the companies, questions were asked about goals and strategies in general, and brand strategy in particular.

### 5.1.2. Current positions

Current positions are, as the term suggests, the actual current business positions of the firm at a given time. However, the business position is the *perceived* position, and since the term is plural, it indicates that this perception may differ between observers. Logically, perceptions should differ if the observer is within the company or from the outside. In general, there should be diminutive discrepancies between the positions. On the other hand, the impact for the firm, if the perceptions differ extensively, may be dire and might result in flawed strategic decisions.

However, the current position is not always easy to measure for the company or external parties. Some companies try to overstate their importance in order to enhance their brands and increase sales, while others take a more humble stance.

Different theories could also be used to describe the current positions for a firm, for example the *value chain* (Porter, 1985) or *value constellations* (Normann and Ramirez, 1994). The former model is often useable in traditional industries, where the value addition could be de-
picted as “strings.” However, when addressing an OSS company, value creation is not as straightforward, which makes the latter model of greater use.

In addition, another issue is “culture,” which reflects the firm’s own view of itself and the environment (Normann, 1975). If the business idea or mission is the formalized (or intended) view, the culture reflects the actual view.

The current positions should also include the views of the firm from the customers, competition and partners, or in other words, how a firm is perceived by the actors in the environment. This would also include government, media, and other participants in the environment. This is important since the view from outsiders might affect how the firm looks at it; depending on its current position or standing in the environment. For example, a firm might want to adapt (to conform to the external view) or reform its external perception.

However, it used to be difficult for a firm to access external views by other actors, unless the company were public. Modern technology, like social media, has made both companies and customers more transparent, which facilitates more contact between different stakeholders. Moreover, for OSS companies, this transparency has been available for a longer time, due to the use of forums and other online chat technologies between them, the customers and users in the community.

Therefore, in this study, the current positions factors that have been studied are the culture and the value constellations. The reason for this is, as discussed above, the difficulties to gauge the position of the company from outside. However, these two factors works like indicators of the actual current positions of the company.

5.1.2.1. Value constellations

In order to study the current positions in OSS companies, the connections between the stakeholders, such as communities, partners, owners, and products are connected. To analyze these connections, a stakeholder/value constellation-mapping model based on Normann and Ramirez (1994) is used. This model was chosen to illustrate how OSS companies add value to their products through their network of partners, customers, and OSS projects, as shown in Figure 5.

![Figure 5: Value constellation mapping, based on Normann and Ramirez (1994).](image-url)
5.1.2.2. Dynamism between visions and current position

A vision should never be fully realized (Normann, 1975), and if a goal is achieved, there should be new ones on the horizon. In other words, there should always be a tension, or a gap between visions and goals compared to the current business positions for a firm to function properly. This is also in accordance to the concept of the changing strategies of a firm (Mintzberg and Waters, 1985).

However, this gap should not be too great for the firm to bridge since that may sow unnecessary tension within the company. If the vision and goals are set too high, or too far ahead in the future, it might foster frustration or a sense of abandonment in the staff or owners.

It might be difficult to judge whether the gap is too great, or indeed too narrow, from the outside. However, for OSS companies there are many second opinions in blogs, forums, and other channels which may draw attention to these gaps. Secondary data from these sources are therefore essential for gaining a more complete picture of the company.

5.1.2.3. A matter of control

As Ouchi (1979) stated, all companies have to employ some sort of control mechanism to make sure the organization moves towards a desired objective (or set of objectives). The three fundamental control mechanisms consist of: a) market mechanism, where price is the determining factor; b) the bureaucratic mechanism, where monitoring and direction of employees are determined by sets of rules; and c) the clan mechanism, where informal rules and rituals determine conformance and membership.

As for OSS vendors, the interaction and handling of stakeholders is vital. External stakeholders, like customers, are prone to look at market factors, which are determined by the offering and price, which the companies regulate through pricing and competitive strategies. Other external stakeholders, such as regulating bodies like governments and stock exchanges, often demand bureaucratic mechanisms in companies. On the other hand, members of OSS communities, which are not formally connected to the OSS vendor, loathe rule sets and need to be handled differently, which conforms to the clan mechanism.

How the companies handle these mechanisms may explain certain choices in how business models are configured, and how this affects other strategies of the company.

5.1.3. Summary of business positions

The business positions consist of visions and current positions that should have a gap between them. Table 5 summarizes the theoretical influences used in business positions.

Table 5: Theoretical influences on business positions.

<table>
<thead>
<tr>
<th>Concept</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business idea</td>
<td>Normann, 1975</td>
</tr>
<tr>
<td>Ideological, imposed strategy</td>
<td>Mintzberg and Waters, 1985</td>
</tr>
<tr>
<td>Business positioning, strategy</td>
<td>Porter, 1980</td>
</tr>
<tr>
<td>Value chain</td>
<td>Porter, 1985</td>
</tr>
<tr>
<td>Value constellation</td>
<td>Normann and Ramirez, 1994</td>
</tr>
<tr>
<td>Control mechanisms</td>
<td>Ouchi, 1979</td>
</tr>
</tbody>
</table>

Business positions also affect operational platforms, since both the current position and strategies influence how the firm uses its resources. In addition, the business positions also have an effect on the offering, mainly through the pricing strategy. Moreover, the strategies and plans also influence the way the company needs to recruit or in other ways expand its resources as seen below in Figure 6.
5.2. Operational platform

In Kindström’s model (2003; 2005) “operative platforms” indicated that a firm could have several different platforms, while in reality it is more of a singular entity. The renamed “operational platform” still contains all the factors from the older model, but includes a number of new ones, and uses a terminology borrowed from the resource-based view (RBV) (Wernerfeldt, 1984; Barney, 1991). The main feature of the theory is the focus on the resources and capabilities that reside inside a firm. Even if companies have similar market strategies (see above), most firms are heterogeneous. In addition, the difference between competing firms may be capabilities that are not commonly available or hard to obtain (Barney, 1991).

The operational platform consists of the resources, or assets, which the firm uses to interact with its environment and its capabilities. The firms’ capabilities are usually divided into four categories of capital: organizational, human, financial, and physical (Barney, 2007).

5.2.1. Organizational capital

The organizational capital consists of the administrative framework, plans, culture, reputation and relationships with the environment (Barney, 2007).

However, the culture, reputation, and relationships are addressed in the business position category of this framework, as shown in Figure 6. On the other hand, one important factor makes the OSS business model framework unique: the management of the community. The framework and routines the firm employs in order to affect the community are important, which other scholars have noted (Dahlander and Magnusson, 2008). Since the community as a whole or via separate project communities, contribute to the firm, the systems for handling these contributions and the community itself should be considered a part of the OSS firm’s organizational capital. Moreover, since there might be several OSS firms that have access to the same communities, the company that handles the relationship best may benefit.

One type of organizational capital is IPR such as copyrighted materials (software, manuals) etc. Even if individuals own the actual copyrights and make the software and manuals, companies often retain the rights for these types of materials. However, tradable IPR, such as patents and commercial rights for copyrights, are considered as financial capital as described below. On the other hand, an OSS firm might just as well conduct business without having any IPR protection at all.

Figure 6: Impact of business positions.
Another type of organizational capital is partner agreements. The reason that these agreements are considered as resources is that they contain specified resources in the form of services offered by a partner, which can be added to the services offered by the firm. The agreement is part of the organizational capital, since the firm can decide to utilize them when needed, while the actual services provided from the partner are part of the offering (see below). The reason why services offered from partners differ from contributions from the community is while the latter contributes for free, the former offers its services for a fee. Therefore, these resources cannot be part of the internal resources of the firm, while the community contributions can. However, the offered services may be bundled to the firms’ offering and thereby add value to the customer.

Strategic and business plans are also organizational capital. They are often an investment in time and resources for a company. The written plan itself is the receipt of that investment, and something that could be shown to investors or owners to verify that the planning has been done — and is therefore a resource in itself. Moreover, the plan often encompasses how the company perceives itself in the market, and sets the foundation for the operational platform.

In addition, the pre-study suggested that brands and branding strategies were of importance. The brand strategy points towards the future, making use of the firm’s mission and vision (Keller et al., 2008). The brand strategy can reflect the business idea, vision and aspired position, and as the pre-study indicated, especially for OSS companies. The marketing aspects of the brands depend on how the firm is positioning and communicating the values of the firm in its brands.

However, branding strategies have traditionally been associated with consumers (Kuhn and Alpert and Pope, 2008), and this has formed the concept that a brand resides in the consumer's mind (Keller, 1993). This suggests that brands are something subjective and hard to measure. Different terminology uses an array of different factors to evaluate and understand how brands function in the consumer's mind. One of the most widely used terms is brand equity (Keller, 2000; Muzellec and Lambkin, 2006; Keller, Apéria and Georgson, 2008; Na, Holland, Shackleton, Hwang and Melewar, 2008). Brand equity is based on the concept that the effectiveness of a brand depends how it is perceived, understood by and affects customers (Keller et al., 2008). The concept of brand equity would indicate that brands should be part of the business positions in the framework. However, since measuring brand equity is troublesome, since brand strategy is a vital part of a firm's marketing endeavor, and because brands are tradable items, brands and finalized brand strategies are an important part of the resources of the firm.

Moreover, the brand equity concept can be translated from the business-to-consumer perspective to business-to-business customers (Kuhn et al., 2008). By extending the concept, the corporate brand equity can be described as the positive difference between how customers and other parties respond to marketing or communication compared with an unknown competitor (Keller, 2000). In other words, brand equity arises when a known brand has unique, strong and positive associations in the mind of the customer (Ibid.).

5.2.1.1. Dynamic capabilities

When Teece, Pisano, and Sheun (1997) introduced the term “dynamic capabilities” they added a tacit dimension to the resources. Dynamic capabilities was created to describe the way the firm adapted to the environment by using managerial skills to adjust the firm’s resources and skills accordingly. The dynamic capabilities are presented in terms of processes, positions, and paths for the internal workings of the firm, external matters and strategic alternatives (Teece et al., 1997).

In addition, Teece also discussed “factors of production” as something available for the entire industry (Teece et al., 1997, p. 516). For OSS companies this would be the source code
for the software, but also the tools for manipulating the source code (operating systems, compilers, programming languages, editors, etc.).

As for strategic alternatives, the capabilities and experiences of a firm make it path dependent, which means that the intangible resources are unique (Barney, 2007).

Eisenhardt and Martin (2000), however, claimed that some capabilities could, or should, be shared across firms in an industry, identified as “best practice.” Of course, these shared dynamic capabilities are not unique for some processes. However, there are a limited number of workable options, and if a firm does not use “best practices,” it could suffer in competition. In addition, some processes, for instance financial statements, are often regulated, which limits the number of ways they can be done. One example of “best practices” is how to make the best use of the undifferentiated factors of production (Ibid).

In modern service-based industry, the competence of the staff is often a very important differentiating factor. The tacit knowledge, i.e. the knowledge that comes from individual experiences and is hard to codify, is something that has become increasingly important. If a company is able to make use of tacit knowledge and translate it to explicit knowledge, it will have an edge on its competitors (Nonaka and Takeuchi, 1995).

A matter of gaining, maintaining and using resources

Through RBV, researchers have gained insight into the firm and with it new knowledge of the inner workings of a company. A firm needs to gain and maintain resources and then utilize them in an effective way to create a sustainable competitive advantage (Barney, 1991; 2007).

The operational platforms also influence the business positions, and not only through “position” as Teece puts it. The strategy may be based on unique resources, and the resources may also limit the number of potential customers (1997).

5.2.2. Human capital

Barney identified several traits for individuals in the firm as human capital, such as training, experience, and relationships (2007). For OSS companies, human capital may be the most important capital of all, depending on the “core competence” of the company (Prahalad and Hamel, 1990). If the firms’ core competence is to produce software or services based on OSS software, human capital is extremely important. In fact, for most of the companies working with OSS, human capital should be considered the prime resource.

One of the insights from the pre-study and the articles was that many firms considered the community, mostly the project community most closely connected to the company, as part of their resources (Dahlander and Magnusson, 2005; 2008). This means that there are operational and managerial aspects that need to be considered when interacting with communities. Dahlander and Magnusson outline some strategies for community interaction, found through empirical case data from Nordic firms (among them Cendidio and MySQL) (2005). In addition, in a later article these authors present three themes on how the same cases make use of their communities. The themes - accessing, aligning, and assimilating - were used to investigate how the firms used OSS within their business models (Dahlander and Magnusson, 2008). These articles are mainly address the innovation aspect of OSS and focus on certain practices in the firms. For instance, they discuss how firms try to influence their communities, for example by assigning the “fun tasks” to the community, while giving the routine tasks to hired staff (Dahlander and Magnusson, 2008, p. 645).

However, there is one aspect they do not cover: recruitment. In some cases, the recruitment process directly involved developers from those communities. This is why the community, or at least the most active developers in the community, should be counted as resources, in the form of human capital.
5.2.3. Financial capital

Financial capital usually takes the form of monetary capital from different financial stakeholders, for instance banks. In addition, retained earnings are considered as financial capital (Barney, 2007).

In this framework, signed customer agreements are considered to be financial capital. A customer agreement is a resource in two ways: first, it stipulates allocation of resources towards the customer, such as goods or services; second, it will yield monetary assets to the company. Agreements give the firms further options for financial capital, for example by taking out loans against the agreements or selling the invoice.

Another form of financial capital is tradable IPR, such as patents and copyrights, or at least the commercial use of copyrights. One subset of IPR is brands, namely corporate and product brands. They are also tradable and have a value on the market, especially corporate brands which can sometimes be valued much more than traditional assets. Balmer and Gray (2003) argue that corporate brands in many ways are the epitome resource for RBV, since the brand affects the whole company. Building brand images and equity takes time and effort, so the value of a brand should be considered as financial capital.

However, assessing the financial value of copyrights, patents, and brands is under debate, especially the latter, since there are different ways of evaluating them (Balmer and Gray, 2003).

One important aspect of the financial capital is the investor or owner. Investors might have significant impact on firms in all aspects, from visions to operative matters such as recruitment and investment in equipment. However, the impact varies between different kinds of investors or owners; public companies, for example, may be less affected by individual investors than the whole market, while a big owner in a private company carries a lot more weight.

5.2.4. Physical capital

In general, the access to raw materials, production equipment, stock of inventories, and sometimes the geographical location of the firm are considered to be physical capital (Barney, 2007). However, for IT companies, raw materials and inventory stocks are not as important for computing physical capital as computers and geographical location. The former is self-evident; the computers are the tool, or production equipment, whereby the firm adds value to its customers. However, the latter has other implications. At a time when the Internet can be reached from almost anywhere in the world, the geographical location for a software firm may not be as important for customer contacts, recruiting purposes, and other cluster effects (Porter, 1998).

Today, however, few physical resources are needed to start-up an OSS firm. Computers are ubiquitous and no longer a significant capital investment, while Internet access is widely available. What is needed is human capital.

Table 6 summarizes the influences used in the operational platform.

<table>
<thead>
<tr>
<th>Concept</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brand strategy</td>
<td>Keller et al., 2008</td>
</tr>
<tr>
<td>Community interaction</td>
<td>Dahlander and Magnusson, 2005; 2008</td>
</tr>
<tr>
<td>Core competence</td>
<td>Prahalad and Hamel, 1990</td>
</tr>
<tr>
<td>Dynamic capabilities</td>
<td>Teece et al., 1997; Eisenhardt and Martin, 2000</td>
</tr>
<tr>
<td>Resources</td>
<td>Barney, 1991; 2007</td>
</tr>
<tr>
<td>Tacit knowledge</td>
<td>Nonaka and Takeuchi, 1995</td>
</tr>
</tbody>
</table>

Table 6: Theoretical influences in the operational platform.
Even if the business position influences the operational platform via strategies and plans, unique resources and capabilities may also affect plans and strategies. In addition, the capabilities of the firm also affect its offering. Experiences and path dependencies may influence the company to be daring or different when it comes to creating offerings for its customers. This is illustrated in Figure 7 below.

Figure 7: Operational platform influence.

5.3. The offering

The offering concerns the products the firm exchanges with its environment. Moreover, the offering is dependent on the other two elements, business positions and operational platform. In the framework, business positions connote interaction with the environment, while the operational platform supplies tools for this interaction. Based on this, the offering is the exchange between the other elements. The offering illustrates how the firm exchanges its resources and capabilities to the stakeholders with which it interacts.

In the framework, the offering contains different factors: what the firm is offering, how the offer is presented to the stakeholders, how the product is distributed, and how the firm gains revenues in exchange for its products. Translated into factors, or sub-models, the offering contains the following: the products offered, the revenue models, the distribution models, and the price models. These factors differ from factors in the other elements, since each factor may have several different ways it depends on markets or customers. Therefore, three of the factors contain “models” instead of describing the concepts one-by-one.

A business model often has different offerings, depending on market aspects, for example segments, type of customer, and volumes. In addition, a firm usually has more than one product that it offers at the same time, with different factors involved.

However, each offering has a limited lifespan. Not only may the products have a limited lifespan; the price cannot stay the same indefinitely. In many cases, each offering is presented

A firm is believed to add value in some form to their stakeholders, as partner, supplier, or customer. This value-adding activity is connoted as a “product” or output from the firm, which can be services and goods in any combination.
with a timestamp, stating how long the offer is valid. When the time is up, the offering is discarded or rendered invalid.

5.3.1. Different views on the offering

Normann argues that a fit must exist between the market segment, the product, and the internal organization (1975). This view is very close to that of this framework. If the offering is not adapted to the market or market segment, it will not succeed. In addition, if the offering is not related to the operational platform, the firm may end up offering products that it cannot deliver.

On the other hand, Kindström sees the offering as an interface between strategic positions and the market (2005). In this view, the offering is the way in which the firm interacts with the market, or even presents itself to the market.

Normann expanded on his view of the offering in 2001. He saw the offering as a method for co-production, where the firm engaged its customers in the creation of value. In addition, offerings are “fixed knowledge,” the result of a process of creation and innovation. However, in order to be competitive, the offerings also need to stimulate learning in the customer. The offering should not be static; with an “open architecture,” the firm has a better chance of staving off competitors.

Normann also concluded that the offering itself is not the “price carrier,” i.e. the unit that determines the actual price. An offering can consist of several different “sub-offerings” with different price tags. The price model, namely, how the company charges for the complete offering, determines the different price carriers (2001). This is the reason why a revenue and price model has been included in the business model.

5.3.2. Offering as an artifact

The offering is seen as an artifact, as noted by Normann (2001). Even if the offering is part of a process, or generates a process for the customer, the offering in itself is an artifact. Another rationale is that processes are a vital part of a firm’s capabilities and are expressed in the operational platform, while customer and partner relationships are handled in the business positions.

In the sales process, or when working closely together with a customer in a joint project, the details are still handled in the other two elements. In addition, there has to be an offering sometime in the process, i.e. one that stipulates the services, products and how much they cost.

In theory, this means that a business model may consist of a limitless number of offerings, each with certain properties. However, having too many offerings outstanding at the same time strains the organization and is therefore not practical.

5.3.3. Offer models

The way in which an offering is presented to a customer may differ between firms and even between different offerings from a single firm; some rely on catalogs (IKEA for example) while others use a network of retailers or resellers, price lists (on paper or on the Internet), or combinations of these.

The offering models are important when studying a company, since they can provide insights into how the firm addresses its customers. It may also be an indicator for how the company perceives its environment.
Some offerings may be composites, consisting of products from different suppliers, as is the case with many offerings today. In order to understand how the offering is presented, the price model would aim to deconstruct it and study each part.

It becomes even more interesting if a firm has several offerings for different customer segments and different ways for those offerings to reach the customers. Studying the difference may give insight into how the firm perceives the different segments, but also how its different customers are used to having offerings presented to them.

5.3.4. The products

In this thesis, the term “products” includes both goods and services. This concept illustrates that products are the output of a company to which the firm has added value. The relationship between service and product business models for the software industry has been described as problematic due to the differences between the different positions. Nambisan (2001) identifies five key issues: intellectual property rights, product complementarity, returns from scale, abstracting knowledge, integrating technology and the connections with users.

In short, products are what the firm offers its customers. It should be noted that the number of products offered by a firm might differ between different customers or segments. The products are connected to the operational platform in the framework. Ideally, therefore, the products offered should mirror the actual resources and capabilities of the firm, or at least to a degree. This research has found that in technologically “heavy” firms, i.e. those with a solid engineering background, the offerings fit the resources and capabilities well. Nevertheless, the offerings present a way for a firm to “test” the market with a new concept or product, for example one not yet fully developed or finished, in order to gauge interest in the environment.

5.3.5. Revenue models

Revenue models are the way a firm charges for the products it offers (e.g. Amit and Zott, 2001). Because revenue models are connected to business positions, they may also differ between firms; in some cases, they might even constitute a primary competitive scheme. This is often the case with service offerings on the Internet.

Revenue models may also differ between different offerings, even if the same products are offered. For example, Microsoft offers products with different revenue models to different segments of customers.

5.3.6. Distribution models

Traditionally, a product was delivered to customers in a way that involved the act of physically moving the product from the site where it was produced to the place where it was sold. Commercial software is often distributed in a paper box, which contains the software product on some kind of media (today mostly DVD-ROM) together with a manual or installation instructions, license agreement and license number. Today, however, most commercial products are available for direct download from the Internet.

An interesting note regarding goods and services is that a service is usually produced at the same spot it is manufactured, and does not need to be distributed. Of course, the consultant might have to be “distributed” to the site where he “manufactures” the service, but in reality, the physical location matters very little. Thus, distribution is more important for goods than for services.

Since the companies in the study mainly offer software products that also include services, the distribution model is primarily concerned with the software product. Moreover, since most
of the included services relate to support, where normally the service is produced at the firm’s premises, the distribution of support is not discussed in detail.

5.3.7. Impacts on the other elements

The theoretical influences for the offering are summarized in Table 7.

Table 7: Theoretical influences for the offering.

<table>
<thead>
<tr>
<th>Concept</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Products</td>
<td>Nambisan, 2001</td>
</tr>
<tr>
<td>Revenue models</td>
<td>Amit and Zott, 2001</td>
</tr>
</tbody>
</table>

Figure 8 shows how an offering may affect both the business positions and the operational platform. In the latter case, it happens every time a customer accepts an offering. The actual offering disappears (unless it is a generic one, such as a price list), but the agreement itself becomes a part of the firm’s resources. An offering may affect the business positions if it changes the way the environment views the firm. For example, when a firm makes an offering in a new market, it gets reactions from customers, competitors and even suppliers as well as other potential partners. In addition, every new customer that accepts an offering is added to the business positions.

Figure 8: The impact of the offering.

5.4. Use of the business model framework

This business model framework is primarily an analytical tool that enables a researcher or consultant to acquire a more holistic view of how a firm operates. The framework presented here is a generic one, which directs the researcher to the theories that inspired it. To be most effective, however, the framework should be adapted to each individual researcher.

By using the business model framework, the researcher may also find certain configurations or archetypes that are successful (or not) in a certain industry. By applying that knowledge,
other firms may improve their business models in order to become more effective. A firm may also find deficiencies in the fit between the different elements that need to be addressed in order to make better use of resources or positions.

5.4.1. Configurations

One important aspect of using a business model is to determine important configurations that can illustrate how a company conducts its business, i.e. what people generally refer to as business models. Miller and Friesen (1994) inspired the usage of “configurations” based on the concept of “quantum states and changes” (1984). Even if the configurations presented in this thesis make use of the same typologies and taxonomies as these authors, the idea is to find certain configurations of OSS vendors that are more sustainable (and profitable) than others, and to describe them.

5.4.2. The cases

Each firm is described based on the elements in the business model framework. The empirical data was collected through semi-structured interviews, using the interview guides found in Appendices 3 and 4. Most interviews were conducted by two researchers on at least two occasions and digitally recorded. In addition, data was collected through e-mail, telephone interviews, and secondary data sources such as annual reports, company websites, and other Internet sources. In the Cendio and MySQL cases, additional material has was to illustrate some examples (Dahlander and Magnusson, 2005; 2008). It should be noted that the author participated in these studies as the Cendio representative.

5.4.2.1. Structure

In order to better structure the data, the cases were organized based on the business model framework.

In addition, within each case there is an analysis, value constellation mapping, which are used to study the brands and how the company is situated in its network of customers, partners and communities.

The business positions, with the visions and current positions, were studied through interviews and secondary data in order to pinpoint the strategies, while the current position was gauged through interviews about the culture, and analyzed through the value constellation. This is illustrated in Table 8, below.

Table 8: Business positions.

<table>
<thead>
<tr>
<th>Business positions</th>
<th>Visions, goals and strategies</th>
<th>Current position</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Business idea or mission</td>
<td>Culture</td>
</tr>
<tr>
<td></td>
<td>Vision and goals</td>
<td>Value constellation mapping (analysis)</td>
</tr>
<tr>
<td></td>
<td>Competitive strategies</td>
<td>Competitors</td>
</tr>
<tr>
<td></td>
<td>Brand and marketing strategies</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Pricing strategy</td>
<td></td>
</tr>
</tbody>
</table>

The operational platform consists of four different types of capital: organizational, human, financial, and physical. As for the physical capital, this element will only describe owned, short-lived assets, such as computers, or leased assets that have not been included. In addition, none of the investigated companies had significant physical assets, so this element was omitted. However, there might be assets, such as land, buildings, or other physical aspects that could impact the firm’s capability to obtain other capital, such as financial capital; therefore,
this element should be present in other investigations. Data for the operational platform was collected through interviews and secondary data sources. The investigated factors are summarized in Table 9 below.

Table 9: Operational platform.

<table>
<thead>
<tr>
<th>Operational platform</th>
<th>Organizational capital</th>
<th>Human capital</th>
<th>Financial capital</th>
<th>Physical capital</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Capabilities, experiences, paths</td>
<td>Community interaction</td>
<td>Annual reports</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>Intellectual property rights</td>
<td>Recruitment strategies</td>
<td>Investors</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Partner agreements</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Finally, the offering consists of four different factors: offer models, products, distribution models and revenue models as shown below in Table 10:

Table 10: Offering.

<table>
<thead>
<tr>
<th>Offering</th>
<th>Offer models</th>
<th>Products</th>
<th>Distribution models</th>
<th>Revenue models</th>
</tr>
</thead>
</table>

5.5. Business models and OSS

When Raymond wrote his essay *The Magic Cauldron* in 1999, he attempted to describe existing business models for OSS, but he also provided two speculative business models.37 His business models focused on the product offered and how revenues could be collected. His descriptions of existing and potential business models are akin to those used in academia from the same time (e.g. Timmers, 1998; Rappa, 1999). However, while the essay is part of a collection of essays, Raymond also implies how the software should be developed as OSS, while articles in academia imply proprietary software without addressing it.

Raymond’s essays (2001) were written from an empirical perspective, strewn with actual examples, albeit personal or based on second-hand sources. He also provides a credible foundation for why direct-sale models, i.e. revenues through proprietary licenses, are not compatible with the hacker culture before presenting his two concepts: use-value funding models and in-direct value sale models. Use-value funding is derived from the logic of spreading or avoiding risks and costs in developing software, while indirect value address services or other aspects derived from the use of software. These concepts were the foundation for the early entrepreneurs of OSS, where only two types of licenses were used, the restrictive GPL and the public domain licenses like BSD and MIT. As a side note, the logic behind the use of restrictive licenses for OSS products were unopposed in academia until 2005, when researchers found that projects with less restrictive licenses attracted more developers than those with restrictive licenses (Lerner and Tirole, 2005).

In addition, Raymond provides prospective OSS entrepreneurs with advice on how to start and maintain an OSS business, with ample examples. This is one of the reasons why some early OSS researchers and observers used Raymond as a starting point and, based on his models, added other existing or potential business models (e.g. Hecker, 1999).

However, later OSS business model researchers and observers also described existing or potential business models in a similar manner as Raymond (see Table 3 above for some ex-

37 The essay is part of Raymond, 2001 and was revised several times before publishing.
amples). In fact, Raymond’s significance for the industry is not exaggerated, since even Red Hat’s subscription model, as described by Goldman and Gabriel (2005) and presented in the Red Hat case above, is a combination of Raymond’s concepts Give Away the Recipe, Open a Restaurant and Free the Software, Sell the Brand (see Table 2 above). In essence, Red Hat redefined the former concept, which was their original business model, re-combined it with its strong brand, and changed their market focus towards enterprise customers. This concept has been emulated throughout the industry at a rapid rate, which both signifies Red Hat’s market leadership and Raymond’s influence in the industry.

However, OSS business model research has not evolved in the same way as business models in academia (Osterwalder et al., 2005). The business models described by Raymond and others can be categorized in the business model chart (Figure 1) as business model descriptions, since they are both descriptive and specific. This corresponds to the first activity of classifying and describing business models (Osterwalder et al., 2005). Where business models in academia have transcended this stage towards listing individual business model components, modeling and testing business model elements and then applying them, OSS business models went directly from the first stage to the last.

This discrepancy depends on three aspects. First, some of the articles on the subject were written by practitioners similar to Raymond (e.g. Stallman, 2002; Perens, 2005; Goldman and Gabriel, 2005). Listing components, modeling and testing have little bearing for practitioners; testing is done in the real world, and faulty models are removed from the industry by market forces. Second, the business side of OSS is relatively less researched compared to other aspects, which could compel researchers to observe, rather than hypothesize or further theorize. Third, with the first and second aspects in mind, the research seems to be focused primarily on application and usability for practitioners (e.g. Goldman and Gabriel, 2005; Perens, 2005; Bonaccorsi et al., 2006).

However, different concepts that are observed should lead to analyses and hypotheses regarding their existence. The more common OSS business models become, the more attractive they should be to researchers, and this will probably create more hypotheses and theories over time. Lately, researchers have started to theorize about OSS business models (e.g. Fitzgerald, 2006; Campbell and Garcia, 2010; Perr et al., 2010).

However, given how OSS business models are conceptualized by observers and researchers, focusing on factors concerning the offering itself, i.e. the product and the revenue model, leads to a limited understanding of how the firm conducts its business. At the same time, other researchers have studied equally important aspects, such as community relations and licensing, with little or no connection to what the company offers.

The aim for this thesis was to provide a more complete view on these companies by combining more factors and making use of established management theories and business model concepts, and based on this create a business model framework that is used to analyze OSS vendors. The framework combines various theories in such a way that different aspects of how firms conduct their business are clarified. It also furthers the concept of business models from offerings to more complete business concepts. The aspiration of the author is that the thesis will further the understanding of OSS vendors, both for researchers and practitioners.

In addition, by using a framework to present the cases, the analysis of the cases has been facilitated; this should make it possible to compare the individual firms, but also provide knowledge that is easier to validate.
6. Case 1: Cendio AB

Cendio was founded in 1992 as Signum Support AB in Linköping, Sweden. Its founders were reportedly inspired by Cygnus Solutions Inc. The business mission was to offer support for free software to local customers, mainly the university and companies in the nearby technology park, Mjärdevi. However, consultancy, consisting mainly of programming missions, became the staple revenue for the company.

Early on, the company decided to try to create packaged products. The first product, Readynet, was an Internet server software with a GUI (Graphical User Interface) aimed towards Internet Service Providers (ISP). It was launched in 1995. The year after, the Fuego Firewall was released, with the same market in mind. Both products consisted of free software, combined with hardware and an advanced GUI. The goal was to create a product that was easy to maintain, even for non-engineers. However, the introduction of other web server software led the company to discontinue Readynet in 1998.

In 1999, venture capitalists invested in the company. At the time, the firewall product and the consultancy business, mainly in embedded systems, was making headway. As a part of the remake, the company changed its name to Cendio Systems. A third business area, server systems, was added to the portfolio and the company grew from about twenty employees to peak at around fifty, opened a second office in Stockholm and appointed a new CEO.

In 2001, however, the company started to feel the recession in the telecom industry, where the majority of its customers were located. The consultancy contracts were terminated and the sales from the server and firewall business decreased. The company decided to spin off the Fuego Firewall product to a joint venture, Ingate Systems, and then sold the rest to the other partner in 2002 to finance the change in the company. Moreover, the Stockholm office was discontinued and most of the staff was laid off, and a fresh direction was to taken to develop and sell a new software product, called ThinLinc.

The first release of ThinLinc was in 2002; since that time, there have been three major releases (Cendio, 2010a)

6.1. Business position

6.1.1. Business idea

The business idea for Cendio is to develop software for server-based desktop computing, and by using OSS with its own proprietary code, to produce robust, adaptive, and price-worthy software for its customers.

38 Early contacts between the founders of Cygnus and the founders of Cendio was confirmed by Michael Tiemann, Vice President Open Source Affairs, Red Hat Inc., in the interview on October 25, 2005.


6.1.2. Vision and goals

Cendio’s long-term vision was to become one of the leading software vendors for server-based computing software. Working towards this vision, the company aimed to work with resellers country-by-country. However, Cendio's new management has changed the goals. The current goal is to become profitable at home before aiming abroad, but even before this new direction was taken, Cendio already had resellers in Europe, United States, Brazil, and Malaysia.

Cendio uses a partnering strategy, with the goal to attract resellers as well as hardware and software vendors. The partners are divided into Technology Partners and Resellers and Distributors (Cendio, 2010b).

Cendio is currently working to establish itself as a preferred software vendor for server-based computing for Swedish health services and municipalities. In addition, the graphics functionality of the product has attracted high-tech companies.

6.1.3. Competitive strategies

Cendio considers OSS as a vast pool of resources that can be tapped in a wide variety of ways. In order to get useful products out of this vastness, a firm needs certain capabilities. These capabilities are usually in the form of specialized knowledge, such as programming and knowledge about OSS licenses (Dahlander and Magnusson (2005).

One of the benefits of this is that Cendio’s development department does not need to be very large; this means lower cost and consequently betters margins. Nevertheless, selling a proprietary product made largely from OSS is not without controversy. Some of the open projects Cendio relies on may not progress in the desired direction, or develop very slowly. This is because development in many OSS projects is based on a voluntary basis and the interest of the maintainer of the project.

For Cendio this means putting its own resources into the projects and contributing with skills, time, and code. The pressure on the programmers to conform to deadlines and follow roadmaps to satisfy the customers (and the company) is a violation of OSS’ credo, which values code quality above deadlines. However, there are forces that have changed this and made projects more structured. Fitzgerald (2006) calls this OSS 2.0.

Cendio’s product, ThinLinc, is a software product based on in-house proprietary code, along with OSS. This mix enables ThinLinc to offer wider functionality and adaptability than its competitors, as seen from both a functionality and development cost perspective. The product offers customers the possibility to combine UNIX and Windows applications on virtual desktops (Cendio, 2010c). In order to better control development, Cendio has been testing new strategies ranging from co-creating a fork 41 to sponsoring a project (Åstrand, 2009; VirtualGL, 2010).

Cendio’s competitive strategy is one of differentiation, together with low-cost development. The product is moderately priced compared to its main competitors with the same functionality, and offers its customers the combination of using legacy hardware on the desktop and combining Linux and Windows applications for the users. The customer may choose to migrate from Windows to Linux applications, thereby reducing licensing fees.

---

41 A fork means that an OSS project is branched and takes on a new direction (for more information, see Raymond, 2001).
6.1.4. Branding and marketing strategy

The company has patent-protected the brands Cendio (corporate) and ThinLinc (product) in Sweden. The corporate brand, Cendio, was created in 1999. The company was relatively well known in Sweden and in the OSS community under its old name, Signum Support, as the firm pioneered Linux in Sweden. The new name became known because the firm was the largest OSS company in Sweden, and among the largest in Europe in 2000. At that time, Cendio was marketed through merchandise, PR, selective advertising, fairs and word-of-mouth. The brands focused on both the corporate brand and the product brand, Fuego Firewall.

Today, Cendio no longer actively promotes its corporate brand; rather, all promotion is centered on the ThinLinc brand. The corporate brand is used mostly for sponsoring events and communities. The marketing is conducted through direct and indirect sales, attending and sponsoring customer-related fairs and events, and occasionally ads in specialized magazines in conjunction with events or fairs. In addition, some effort is given to PR, mostly aimed at national IT magazines such as ComputerSweden and TechData, and through the company website (cf Åsblom, 2010; Agfors, 2009).

6.1.5. Pricing strategy

Cendio’s price strategy is to be a mid-price vendor, offering high security, robustness, and specialized functionality at a lower price than the market leaders. The reasoning for this strategy is that the product delivers more value than free OSS products and low-end competitors, and that development costs are lower than with proprietary software, since the product is mainly based on OSS.

6.1.6. Culture

Cendio has had a turbulent past with expansions and layoffs and a strained economy. The present company is small and tightly knit, and all work together to achieve its goals. However, the goals may have been set too high, and the new management (since 2009) has focused on making the company profitable.

The firm has been working with OSS for a long time, and the philosophy and mindset is based on the same creed. Openness and frankness is something that the old and new CEO frequently communicate, even among customers and partners. In addition, the content is adapted based on inputs from customers, either directly or through a customer user group.

Since Cendio’s in-house developers are active in OSS projects, the same tools and principles, such as rapid prototyping, are used for the in-house proprietary development as well.

6.1.7. Value constellation mapping

To analyze how Cendio and its partners add value for customers, a value constellation model has been used, based on Normann and Ramirez (1994) and stakeholder analysis. The mapping for Cendio is straightforward, as shown in Figure 9 below. The firm is quite small, and offers just one product. Its main business is in Sweden, where stands for about 90% of all sales. The sales partners outside of Sweden are usually firms that have contacted Cendio asking to sell the product in their respective country or region. The technology partners are connected either to hardware or software integration or certification.
ThinLinc uses a variety of OSS, which is covered through a complex system of licensing (Cendio, 2010c). However, given the number of in-house developers, Cendio is only able to be active in a limited number of projects. The two primary projects to which Cendio actively contributes are TigerVNC and rdesktop.

**TigerVNC**

The key OSS component of ThinLinc is Virtual Network Computing (VNC). This software was originally proprietary, but a hybrid version, RealVNC, was created in 2002. RealVNC releases older, less-capable versions to OSS (under GPL) while retaining newer versions as proprietary (RealVNC, 2010). Cendio used and worked with TightVNC, a version of VNC designed for lower bandwidths. After four years, it was clear that development in the project was not going the desired direction. Consequently, a new VNC fork, TigerVNC, was created in 2009 together with two developers in similar projects (Åstrand, 2009).

**Rdesktop**

Rdesktop is an OSS client for Windows Terminal Services, a service that allows a user to remotely connect to a Windows server or client. Cendio is actively contributing to the project, and has developed a module for rdesktop, called SeamlessRDP, which further enhances the functionality for ThinLinc (Cendio, 2010d).
VirtualGL

A new strategy for Cendio is to sponsor OSS projects. Cendio is currently an active sponsor for the project VirtualGL (VirtualGL, 2010a). VirtualGL enables server-based clients to run graphical applications with higher performance on the servers, which for example makes it possible to run CAD software on a server, rather than on expensive desktop computers. VirtualGL is mainly a one-programmer project, but with a reputable and well-known career (VirtualGL, 2010b).

Other projects

Over time, Cendio has contributed to other projects, for example PulseAudio, but these contributions have been limited in time and scope (Ohloh, 2010). Cendio has had a low profile in these projects, with less attention given to the company name.

6.1.7.2. Partners

Cendio has three different types of partners: technology partners, distributors and resellers.

Technology partners

Cendio has technology partners in three areas: development, software integration, and hardware integration.

The only development partner is Red Hat, which is part of the TigerVNC project. The companies support the project through code contributions and project management.

However, Red Hat is also a partner concerning software integration, together with CodeWeavers, Nordic Edge, and Novell. The software integration partners are all software vendors, and their software can be used in ThinLinc. Red Hat and Novell offer commercial Linux distribution for which ThinLinc is certified. In addition, ThinLinc has several integration tools for Novell software, since it is widely used in schools and municipalities (Cendio, 2010e).

CodeWeavers’ product suite CrossOver can be used to run Windows applications on the Linux server used for ThinLinc as a substitute for a real Windows Server. By using CodeWeavers, a customer who owns Windows applications licenses, but not any operating systems licenses, can use these applications with ThinLinc (Cendio, 2010f).

Nordic Edge is a Swedish firm specializing in security solutions. One of its products has been adapted to work in conjunction with ThinLinc, offering higher security for the single sign-on solution in ThinLinc (Cendio, 2010g).

Cendio is also working with hardware vendors, mostly for thin clients, and is partnered with two manufacturers, IGEL and Fujitsu Siemens Computers (Fujitsu). IGEL is a German manufacturer of hardware thin clients with support for ThinLinc features (IGEL, 2010). Fujitsu also offers hardware for thin clients (Fujitsu, 2010). These clients can be installed with the eLux operating system, which is developed by UniCon (UniCon, 2010). eLux can be used for other thin clients and is the preferred choice for ThinLinc (Cendio, 2010h).

Distributors and resellers

Before 2009, Cendio was working towards a global presence. In this effort, the firm attracted interest from different parts of the world. In general, Cendio was contacted by the interested party and, based on that, an agreement was signed. Cendio also offered training and support for its resellers abroad, for example training at their location. For instance, Cendio sent staff to Brazil to certify resellers and customers. At this time, the short-term goal for Cendio is to become profitable, which means that the focus has changed towards the Swedish market. However, the existing resellers abroad are still supported.
For Cendio, there are two different types of reselling business partners: distributors and resellers. In general, distributors are more committed to ThinLinc, and are certified to sign offers, manage, and support resellers in their appointed areas. Resellers, on the other hand, may not be interested or capable of managing and supporting other resellers, and therefore have the opportunity to be managed by a distributor, if one exists for their market, or directly by Cendio.

At this time, there are three distributors, covering Brazil, Central and Eastern Europe, along with an undisclosed number of resellers, eight of which are directly managed by Cendio. The company’s current focus on the Swedish market is clear, since five of the eight resellers are located in Sweden and the majority of those are systems integrators (Cendio, 2010b).

Even though Cendio has a partner strategy, the firm still has to attract customers of its own. The reasons are many, including large portfolios, lack of focus and different agendas for the reseller. The size of the reseller is often connected to the size of the product portfolio, which makes it harder for individual salespersons to focus on niche products. The management of the sales efforts is in the hands of the reseller, and the agenda for that company is hard for a small vendor, like Cendio, to affect. In addition, product vendors, through sales and market campaigns, often affect larger resellers, something that Cendio lacks the resources to duplicate.

Smaller resellers, while having a smaller portfolio and more focus on ThinLinc, may lack the capability to handle certain customers. Since Cendio has a mix of large and small resellers in Sweden, the firm realized that it had to attract customers in the initial phase, and then turn them over to a larger partner in the implementation phase.

6.1.7.3. Owners

Cendio is privately owned; its employees and their families own some of the firm, but a private angel investor owns more than fifty percent. He is the chairperson of the board of directors, and very influential in the firm. Thanks to his reinvestments, about twice each year since 2003, Cendio is still an active company. In 2003, he initiated changes in the firm’s goals. Since then, Cendio has failed to earn a profit.

6.1.7.4. Customers

Cendio’s main customers are Swedish municipalities, in the areas of school systems and healthcare. This has been the focus for Cendio since 2003, since the product offers cost savings for its customers. Since 2007, the focus has been on healthcare services, where the security aspect of the product makes it competitive.

However, since the introduction of TigerVNC and enhanced graphics functionality, a new category of customers, manufacturing, has been added. These customers are attracted to potential cost savings on the workstation side, since the product offers high graphic capabilities from the server to the desktop computers. In addition, since all processing and data is done on the server, there are security benefits as well (Åsblom, 2009). This segment is on the rise, which may benefit Cendio in both the short and long run.

Cendio’s customers are not participating in any Cendio-sponsored OSS projects at this time, since most of the customers are not interested in software development. To better solicit the opinions and proposals of its early customers, Cendio urged them to found the ThinLinc User Group (TUG) in 2006 (ThinLinc User Group, 2010). TUG is independent, but Cendio has sponsored the online forum and past events.

6.1.8. Competitors

Cendio competes in a highly monopolized market. When Cendio started in 2002, there were two major vendors, Citrix and Tarantella, with Microsoft offering limited services. Today,
Microsoft is the largest vendor, encompassing most of the market, while Citrix has been marginalized and pushed towards the other end of the industry. Tarantella was purchased by Sun Microsystems, and its product is now part of the SunRay concept, which includes hardware and software (Oracle, 2010).

In addition to the main vendors, there exist hundreds of smaller, localized vendors offering different commercial solutions. In the Linux market, the Italian company NoMachine’s product NX is perhaps the most widely known (Nomachine, 2010a).

There are also OSS competitors to Cendio; a popular software is the Linux Terminal Server Project (LTSP) (Linux Terminal Server Project, 2010), but there are several other projects devoted to server and computer-based VNC (IPinfo, 2010).

The growth of virtualization is also a threat to Cendio, since one of the latest developments is “desktop virtualization,” also known as Virtual Desktop Infrastructure (VDI). Even if Cendio is well-positioned, large vendors like Microsoft, VMware, Red Hat and Novell are starting to develop products as well. The last three have traditionally worked with servers, but are moving towards the desktop through VDI (Rooney, 2006).

A recent phenomenon is the web application business, also known as Software as a Service (SaaS), Web 2.0, or cloud computing (Holloway, 2010). This means that users get access to applications through the Internet using their web browsers. The technology is hyped, and is believed to be the next “big thing” (Gartner, 2008). Products based on this technology are competitors to Cendio today, or may be in the near future. Large vendors in this industry are Amazon, Google, IBM, Microsoft, and Salesforce.com (Saasbuzz.com, 2010).

6.2. Operational platform

6.2.1. Organizational capital

6.2.1.1. Capabilities, experiences and paths

Since its founding in 1992, Cendio has worked with business concepts related to OSS, such as consulting, products, and training. One of the capabilities needed is the ability to understand how a particular OSS can work together with ThinLinc, not only on a technical basis, but also with respect to licensing (Dahlander and Magnusson, 2005). The capability to create “good” source code is essential, which has been proven through peer review (e.g. when a developer is allowed to contribute to a project).

Product development in Cendio has followed a distinctive path. Cendio has always used hybrid licensing schemes. In the beginning, the solution was to bundle software, usually a mix between OSS and in-house, with hardware and support. The customer was charged for the whole package, where the support contract was the only thing that could be renewed. However, computer hardware becomes obsolete very rapidly, and this was a big issue for Cendio and its bundled hardware products. Due to the rapidly decreasing value of the hardware, Cendio was forced to limit its stock, which sometimes negatively affected delivery times.

When Cendio became software-only, gaining higher margins through lower cost per product sold, the path was still followed: a hybrid license, and a mix between OSS and proprietary, in-house software. Even if the company did not offer hardware, its software was still tested extensively on different kinds of systems.

Cendio has always offered training in OSS, particularly with regard to Linux and other software, and used to be one of the premier Linux education companies in Sweden. When Cendio focused on ThinLinc, this legacy was utilized to create certification courses for the product, which each reselling partner had to attend. In general, the reaction to this training has been very positive, even when held abroad in Brazil.
One experience with resellers forced Cendio to change its channel strategy. Since most re-selling partners usually have a wide array of products and services to offer, it is hard for a small vendor like Cendio to gain attention. Therefore, Cendio has developed a “test installation” process, which offers the product and installation services directly to end customers for a limited period of time. If the customer wants a full installation after the trial period, they are turned over to one of the resellers.

6.2.1.2. Intellectual property rights

ThinLinc is a software bundle consisting of both OSS and proprietary software, where Cendio retains the copyright of the proprietary parts. Cendio also retains the copyrights of its training material for the partner certification process.

There has been some controversy surrounding ThinLinc and its licenses, since the packaged product consists of both open and closed software, particularly in regards to VNC, which is a vital part of the solution. Since VNC is using GPL, it has been brought to Cendio’s attention that ThinLinc might be violating the license. However, the company maintains that there are no license issues, because of the nature of how the software applications communicate with each other. Cendio has studied the effects of licensing all proprietary parts to OSS, but no decision has been made.

The corporate brand, Cendio, is a registered trademark in Brazil, Sweden and the United States, while the product brand, ThinLinc is a registered brand in Benelux, Brazil, Denmark, Germany, Finland, Norway, Sweden, and the United States.

6.2.1.3. Partner agreements

In order to become a reselling partner to Cendio there is a certification process, called ThinLinc Professional, with education and tests for technicians developed and conducted by Cendio. This is to ensure that the partner was able to install and maintain a ThinLinc system. Some of the early resellers were not comfortable with Linux systems, so several of the old training courses were adapted in order to get non-Linux technicians proficient enough to be able to get the certification.

In addition, since Cendio was a new software vendor, it was important that the management and sales personnel at the reseller were acquainted with Cendio, the management team, and the product. Therefore, a one-day introduction course was conducted in an informal atmosphere.

However, with the introduction of foreign partners, the process has been simplified and the requirements made less strict, since Cendio is capable of delivering support to resellers without certified technicians.

6.2.2. Human capital

6.2.2.1. Community interaction

ThinLinc is a composite product, composed of different software. The core software, VSM, is proprietary, which means that the firm does not have a designated OSS project for the product. Instead, Cendio works with other OSS communities, such as TigerVNC, rdesktop, and VirtualGL, which provide code for other parts of the product.

Some of the Cendio’s employed developers have gained a positive reputation over the years, and some of the current developers are highly rated in their areas of expertise. For example, Cendio’s chief developer was part of the OSS project team for TightVNC, but founded the fork TigerVNC together with developers from Red Hat and VirtualGL. TigerVNC is a small project, with a few active developers. This is the risk when forking a project, since both the old branch and the new compete for the same pool of developers. However, Cendio is in
control of the project, since its chief developer is the project maintainer and is sponsoring other developers.

The OSS project rdesktop is a larger project, with low activity, where Cendio’s developers are the most active contributors. In addition, the chief developer of Cendio shares the role of project maintainer (Sourceforge, 2010c). On the project’s website, Cendio is listed as one of the companies able to provide professional support for rdesktop (rdesktop, 2010).

Cendio is active in other projects as a contributor or a sponsor. In important projects, such as TigerVNC and rdesktop, the corporate brand is exhibited and the company actively sponsors the project or main developers. In other projects, the company profile is lower, depending on the time and effort of the contributions.

In addition to OSS projects, TUG was initiated in order to channel requests and reports from customers directly to the company. There were events, where the new version of the product was presented together with partners, the last one being in 2008. Lately, however, activity has waned and been reduced to a few discussions in the forum. On the other hand, Cendio is planning to increase these activities again, when the situation stabilizes in the company.

6.2.2.2. Recruitment strategy

Since its founding, the typical employee at Cendio has been a systems operator or a developer. The company has always relied on its close relationship and proximity to the university for recruitment. The usual standard for recruitment has been part-time or summer job employment, in addition to final theses work. Moreover, Cendio has been very active in local computer societies, such as Lysator, and has recruited prominent members over the years.

Cendio's focus has always been on OSS programming, and since it has worked with OSS since 1992, some of its developers have made a name for themselves in various OSS communities. Around 2000, Cendio was the largest Linux-oriented company in Sweden, which added to its reputation. In addition, this brand reputation, together with well-known employees, made Cendio an attractive employer for developers interested in programming.

6.2.3. Financial capital

Cendio is not a profitable company and has been dependent on capital injections since 1999. This situation has always limited the firm’s capabilities, especially in marketing.

Cendio owns two brands, the corporate brand Cendio, and the product brand ThinLinc. The product brand is a registered trademark in Sweden, Benelux, Norway, Denmark, France, and Germany, while the corporate brand is only registered for Sweden (PRV, 2010).

6.2.3.1. Annual reports

Cendio has not been profitable in the last 10 years. Since the new direction in 2002, sales have fluctuated with costs still higher than income. The last annual report shows the effects of the financial crisis with a slight decline. Table 11 below summarizes the financial statements from 2007 to 2010.
Table 11: Summary, financial statements (Affärsdata, 2011).

<table>
<thead>
<tr>
<th>Financial period:</th>
<th>201001 - 201012</th>
<th>200901 - 200912</th>
<th>200801 - 200812</th>
<th>200701 - 200712</th>
</tr>
</thead>
<tbody>
<tr>
<td>Employees:</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Sales (KSEK):</td>
<td>4 404</td>
<td>3 888</td>
<td>4 444</td>
<td>3 041</td>
</tr>
<tr>
<td>Change in Sales (%):</td>
<td>13.27</td>
<td>-12.51</td>
<td>46.14</td>
<td>-7.00</td>
</tr>
<tr>
<td>Result (KSEK):</td>
<td>-2 111</td>
<td>-5 343</td>
<td>-4 959</td>
<td>-5 698</td>
</tr>
<tr>
<td>Assets (KSEK):</td>
<td>5 117</td>
<td>6 563</td>
<td>7 271</td>
<td>7 238</td>
</tr>
<tr>
<td>Solvency (%):</td>
<td>25.03</td>
<td>25.48</td>
<td>41.49</td>
<td>27.29</td>
</tr>
</tbody>
</table>

The efforts at cost reduction and a focus on sales in Sweden have clearly had effect, which is noticeable in the financial statement of 2010.

6.2.4. Investors

The company is privately held, where the founders, current and former staff have minority posts. The majority owner is an elderly angel investor, with a long history of founding and investing in small businesses of different kinds. He has been the sole investor, injecting capital twice a year to keep the firm afloat. He is also chairman of the board, with full strategic and operational insight. Moreover, in 2009 he initiated the change in goals, seeking profitability in the short term, rather than long-term expansion of market share.


6.3. Offering

6.3.1. Offer models

With ThinLinc Version 3.0, Cendio offers the product directly from its website, or through resellers. The prices on the company website are target prices, and the resellers are rewarded through discounts on the official price.

The product can be downloaded and installed for up to ten concurrent users, free of charge. For more than ten users, the company offers a subscription license, which includes a perpetual usage license.

In Sweden, Cendio does offer test installations directly to end users, mostly in the form of governmental organizations like schools or health care services. If the customer is interested in purchasing the product, he or she is referred to one of Cendio’s resellers, who finalize the deal.

The pricing strategy can be described as "mid-price", where the price is per concurrent user and year (Cendio, 2010i). It is an easy-to-understand model, compared to Citrix and NoMachine who charge for the server and clients separately (Madden, 2008; NoMachine, 2010b).

6.3.2. Products

Cendio offers ThinLinc through subscriptions, which include upgrades and basic support for a term of one or three years. The subscription is per concurrent user (Cendio, 2010i).

In addition, Cendio offers extended support to ThinLinc, which can be obtained separately from the subscription agreement. The support agreement extends the basic support included in the subscription, with guaranteed response times and services that are more extensive. The support agreements are valid for a limited number of incidents (Cendio, 2010i).
Moreover, Cendio also offers certification courses for customers and partners. There are two courses, ThinLinc Linux and ThinLinc Professional. The former is the prerequisite for the latter, but if the technician has Linux or Unix experience, ThinLinc Linux is not needed (Cendio, 2010j).

6.3.3. Distribution model

The product is downloaded directly from Cendio or its distributors, and various other websites. In addition, it is also distributed via installation by Cendio or its resellers. However, since the downloaded version is limited to ten concurrent users, the license files needed are distributed directly by Cendio or its resellers. Support services are delivered through e-mail, phone or via remote connection. Training is either provided at Cendio’s premises in Linköping, Sweden, or at any location specified by the reseller or customers.

6.3.4. Revenue model

Cendio earns revenue from three sources: subscriptions, training, and support services. The primary income comes from subscriptions, where customers are charged per concurrent user. ThinLinc has a monitoring system, which allows the system administrators to gauge the need for licenses. Moreover, there are soft and hard limits which provide some flexibility for the customers (Cendio, 2010k).

The second largest revenue stream comes from services, i.e. support agreements and test installations. The support services are relatively new and have not yet reached their full potential.

The training revenues are intermittent, depending on the demands from customers and partners. The company offers the courses on a regular basis, but the courses are only given if there is the minimum number of attendants. In addition, some training is performed outside of Cendio. In these cases, the customer or reseller is charged extra for the additional time and expenses incurred.

CodeWeavers, based in Saint Paul, Minnesota, was founded in 1996 as a consultancy firm. The company was refocused around the OSS project Wine in early 1999. The Wine project allows software written for Windows to run on Linux and other UNIX operating systems (WineHQ, 2010a). CodeWeavers obtained venture capital funding to support a business model encompassing the project. At about the same time in 1999, Corel, a large software vendor, contracted CodeWeavers to work with the Wine project in order to bring its office suite to Linux (WineHQ, 2010a).

Corel decided to discontinue its Linux efforts and withdrew its support for Wine in 2001. CodeWeavers decided to create its product, CrossOver Office, which was released in 2002. CodeWeavers is working closely with the Wine project, and has recruited most of its employees based on their performance in the project. For instance, the Wine maintainer (or project leader) Alexandre Juilliard is the chief technology officer at CodeWeavers (CodeWeavers, 2010a). All code that is developed by CodeWeavers is released back to the community, with the exception of minor scripts. However, some of the code does not make it into the project, specifically that targeting particular problems or workarounds.

CodeWeavers offers three products, CrossOver Mac, CrossOver Games and CrossOver Linux (CodeWeavers, 2010b). The products use a proprietary license, and are able to do this since Wine uses the LGPL license while the other components use BSD-style licenses, which allows for proprietary licensing. However, the proprietary parts, mostly installation and configuration utilities, are a very small part of the code base, approximately 5% of the total code. Moreover, management has made it clear that they are 100% for OSS and if they could make a living out of consultancy, they would discontinue the proprietary part of the product.

Aside from the products, which are offered with support and upgrading services, CodeWeavers also offers a number of services, ranging from consulting services for CrossOver, redistributing software using CrossOver and porting software from other operating systems.

The firm has grown organically since 2002; recruitment has predominately been through the Wine project, with the exception of some key personnel. Financially, the company has been profitable, but with a small profit margin, which has somewhat impeded strategies and investments.

7.1. Business position

7.1.1. Business idea

CodeWeavers’ business idea is to develop software that makes it possible to use software normally running only on Microsoft Windows on Linux and Mac OS X. This is to facilitate migration between operating systems. The mission statement has changed from an initial focus on Linux to encompass Mac OS as well.

The company’s mission statement consists of three sentences:
To transform Mac OS X and Linux into Windows®-compatible operating systems.
To help our customers leverage Windows technology on non-Windows operating systems.
To promote the growth of Free Software by supporting and extending the Wine Project (CodeWeavers, 2010c).

7.1.2. Vision and goals

An important vision for the company is to be able to fulfill its original mission, which was to help Independent Software Vendors (ISVs) to port their products from Windows to Linux and
Mac OS as consultants, rather than as software vendors. However, since 2002, when the company became a software vendor, this vision has not been in focus. However, the upturn of the Mac OS market and the demand for Windows software to run on Mac has recreated this long-term vision, at least in the mind of the founder and CEO. As of 2010, the Wine services portion of the business continued to grow, and now accounts for nearly 25% of corporate revenue.

However, the current situation for the company has forced the management team to focus on day-to-day operations.

“We are very much focused on opportunities that will bring us revenue in the immediate sense. We are what we would call a “coin-operated” company and sometimes this hampers our ability to do long term strategic planning, because we are still not out of the mode of living from paycheck to paycheck.”

CodeWeavers has a planning window of about six months to a year. In 2007, the company hired a Vice President of Sales. Even after a short time in the company, he created goals for the sales department, which was something new for the organization.

An unarticulated goal, but one that is on the minds of the executives, is to achieve a stable revenue stream that allows them to plan and widen their horizon from months to at least one year or more. By 2010, this goal was nearly reached. Despite the generally weak economic climate since 2008, CodeWeavers remains profitable and has continued to increase sales, as well as add personnel.

7.1.3. Competitive strategies

CrossOver is directly based on the OSS project Wine, but with additional installation scripts and other integration features not present in the open project. The paying customer also gets access to professional support.

In comparison to other companies like TransGaming utilizing the Wine project in their products, CodeWeavers clearly has the community’s support. However, the company (and companies like TransGaming) is also dependent on the maturity of the Wine project.

We are sort of caught in a sort of gray area at this point; Wine has got a lot better but a lot better means that it still only runs maybe 50% of all applications that are out there, and even those 50% will have a fair amount of warts on them. It is clear for us that if Wine were the sort of technology that would run 95% of all Windows-applications right out of the box that is a complete scene-changer in terms of the technology landscape. […] The question is how quickly we can get to that point. It is a very difficult technology to develop in; it requires a lot of time and effort. We just need more bodies, and that means more cash flow or a larger Open source community working on that technology.

The recent success of Apple has resulted in an increase in Mac sales and a greater interest for companies to port their software from Windows to Mac OS. This has presented great opportunities for CodeWeavers to retain its original plan for the company.

7.1.4. Branding and marketing strategies

The company uses two brands, CodeWeavers (corporate) and CrossOver (product). CodeWeavers is a registered trademark in the United States, while the product brand is not protected. The company does minimal marketing for the product brand; the strategy is to target well-known bloggers with free copies for reviews. The firm is actively looking for media buzz about its products.

In the past, CodeWeavers visited fairs and events connected to Linux, but these activities ceased after the company released its Mac product. Instead, the main marketing efforts are

---

centered on the corporate brand, issuing press releases about new product releases and new versions of Wine. In addition, the company publishes humoristic press releases, such as the CEO naming himself as the “employee of the month” for the 175th time (CodeWeavers, 2010d). The company also makes use of social media such as blogs, Facebook, and Twitter (CodeWeavers, 2010e; Facebook, 2010; Twitter, 2010).

7.1.5. Pricing strategy

CodeWeavers’ price strategy is to be a low-price vendor. The product competes with the Wine project, and also with native Windows installation (in the form of dual-boot). This ensures a negative price pressure for the product. Even if the product is priced much lower than its commercial competitors’ product in the Mac market, the product functionality may not compare in some areas, which also lowers the price.

The company is less hopeful about the Linux market, partly due to the lack of customers, and partly because the average Linux user is more technology-inclined and less interested in paying for the product. Since the introduction of the Mac product in 2007, product sales of CrossOver have moved from about 50/50 to 65/35 between Mac and Linux. Linux remains an important core business, but its importance is not as great as it once was.

7.1.6. Culture

CodeWeavers’ management, marketing, system administrators, sales, and technical support personnel work in the main St. Paul office. However, the majority of the developers – who are located in about ten different countries worldwide – work from their homes, making a “traditional” company culture difficult. The main form of communication is via chat, so the chat clients are always online, and the company uses IP phones.

The different time zones are still an obstacle for this kind of communication and in certain cases, such as during product releases, coordination within the company is hampered due to the disparity of locations.

However, the different time zones may also have their benefits. For example, a severe bug was discovered one day before a release; since the problem could be transferred throughout the different time zones, a solution was found and implemented just hours before the official release.

The company’s employees are bonded together in their belief in OSS. This bond starts with Jeremy White and Jon Parshall at the top and reaches down throughout the entire company, and is even addressed on the company website. This and the sense of openness in the firm, including candid reporting on the financial situation, are the most important factors that drive developer loyalty to the company, despite shortages of funds and even delayed salaries. This was particularly true in the “lean times” between 2002 and 2006. The same corporate culture has continued, however, even as the company’s financial fortunes have improved.

---

44 Dual-boot means that a computer may be installed with more than one operating system. This is common for Linux computers, but since Apple moved to Intel processors all Macs come with a tool called “Boot Camp” for dual booting with Windows operating systems, see: Apple (2010) Built for compatibility. (HTML) Available: <http://www.apple.com/macosx/compatibility/> (2010-11-08).

45 An IP phone makes use of the Internet instead of normal telecommunication cables, and thus international calls cost the same as local calls (if there is any cost involved at all). One example of this technology is Skype; see more at Skype (2010) Free internet calls and cheap phone calls – Skype. (HTML) Available: <http://www.skype.com/intl/en/home> (2010-11-10).
7.1.7. Value constellation mapping

CodeWeavers’ value constellation map is straightforward, as Figure 10 shows. This is because the company works exclusively with one OSS project, Wine, and has no technical partners. The reseller situation is more complex, because the company has three products with significantly different customer focus and markets, which complicates the reseller situation.

![Figure 10: Value mapping for CodeWeavers.](image)

7.1.7.1. Communities

CodeWeavers works almost exclusively with the Wine project, which is an active community. The community not only contributes to the code, but also submits bug reports and provides other kinds of feedback.

Additionally, CodeWeavers employs the most skilled and dedicated developers directly from the Wine project. Since the company’s CTO is also the maintainer of the Wine project, he personally reviews all code that comes to the project. Through this, he becomes aware of those coders who are doing good work, who are later approached by the company with an offer of employment. The result is that most of the employed developers have been recruited from the project, and most of the code developed by the company is fed back to the project, but not necessarily accepted, as the following quote from Jeremy White shows:

> I think Alexandre has successfully eradicated the last line of my code in Wine (and he stubbornly rejects my patches, too), so nothing is Wine is anything I have built (Slashdot, 2010).

However, the company feels that the Wine community is not large enough, and this worries management. If the company enlarged its business, the lack of developers would be a problem. As of 2009, the company began actively recruiting university students to work as Wine development interns. Several of these students developed sufficient aptitude with Wine to be brought aboard as full-time developers. The company has generally had very good luck working with younger developers who have the spare time and the aptitude for trying out newer technologies.

CodeWeavers has worked with other projects at different times, for instance the Loki installer, but only for a limited time (CodeWeavers, 2010i).

7.1.7.2. Partners

CodeWeavers operates through distributors, strategic partners, one aggregator, and resellers, including Value-Added Resellers (VAR). VAR offer first-line support and installation services for the company’s products.
For the Linux product, there are three regional resellers, one each for Asia, India, and the United States. In addition, the company is working together with four strategic partners, among them Cendio, which also act like resellers or VAR.

For the Macintosh and the Games products, the company has contracted RedFrog, which is an aggregator that packages software and documentation into a box with a DVD-ROM or CD-ROM. These are sold through the distributors Navarre and Douglas Stewart in North America. Through Navarre, the product is shipped to retail chains like Best Buy, CompUSA and the Apple Stores (Navarre, 2010). Moreover, the educational distributor Douglas Stewart has moved the product to campus bookstores (Douglas Stewart, 2010). Recently, however, due to consolidation within the retail space, CodeWeavers is no longer in direct consumer retail outlets other than some smaller regional niche outlets. The other Mac distributors are responsible for Australia, Canada, and Japan.

7.1.7.3. Owners

The company is held privately, with Jeremy White holding the majority post, and the board of directors with a significant share as well. Some of the employees and families also own shares.

Venture capitalists and other investors have approached the owners, but these contacts have never led to any deals. The managers clearly stated that they are not interested in spending the time and effort needed to create an agreement, and they feel that most investors are worried about patent or copyright issues in addition to the lack of their own intellectual property. It should be noted, however, that recent changes in U.S. software patent laws might make it easier for the company to secure funding (SFLC, 2010). While company management is not actively seeking such deals, it is open to them.

7.1.7.4. Customers

Even if the Wine project is free to download and use, the main customers for CodeWeavers are private or business users. The number of users buying the product was what saved the firm when it had to reorganize in 2002. However, on the Linux “side” of the community, the users are more likely to download the free version than pay for CrossOver, while Mac OS users are more likely to purchase the product, preferably in a box, than download the free version.

CodeWeavers’ products and services are targeted towards customers using Linux or Mac desktop systems, and who are interested in using Microsoft Windows applications on those platforms. The company offers its CrossOver product and services related to it.

The new opportunity with Mac OS through a retail channel has increased the branding opportunities for the company through the boxes of software. In addition, Mac OS users seem prone to use the support services, thus adding more contacts for CodeWeavers.

There is also an opportunity for paying customers to suggest or direct CodeWeavers towards the applications they want supported. CodeWeavers’ original mission may return, when ISVs want to port their applications from Windows to Mac OS. This market is on the rise.

The company does not disclose its business customers, but the managers mentioned well-known entities like Cisco, DreamWorks, Pixar, Disney, and Google. Other customers, for example, include higher educational institutions. In summary, CodeWeavers’ customers are not concentrated in any specific industry; rather, they come from a diverse array of industries, markets, and regions.

7.1.8. Competitors

Aside from the Wine project itself, the competition for CodeWeavers has risen since virtualization has become more popular in recent years. The products for virtualization offer most of
the features in CrossOver, but often at a much higher price (and often lower performance). Some examples of competitors in this area are Parallels, VMware, and the free Virtualbox from Oracle. Where Parallels and VMware are proprietary software, Virtualbox is OSS and uses some of the Wine libraries.46

The most prominent commercial competitor that made use of the Wine project’s code base is Cedega from TransGaming (Cedega. 2010a). Before the project changed the license to LGPL in 2002, TransGaming downloaded and based its proprietary product on it and decided to close the source, thus contributing less to the project (Miller, 2004; Cedega, 2010b). Nevertheless, while CodeWeavers focuses on office applications, TransGaming is focused on taking PC games to the Linux operating systems through Cedega, and to the Mac platform via Cider (TransGaming, 2010).

Microsoft, the vendor producing the Windows operating system, which the Wine project is trying to emulate, is of course an important competitor. If Microsoft decides to fully take its applications to the Linux and Macintosh platforms, CodeWeavers and its products would become obsolete. Even if this scenario has been considered more or less impossible for a number of years, Microsoft seems to have moved into this (feared) direction, if somewhat slowly. There is also a risk that Microsoft would try to prosecute CodeWeavers for eventual IPR infringement. The agreement between Novell and Microsoft does not protect the Wine project (and thus CodeWeavers) from this threat (Paul, 2010). However, CodeWeavers owns legal research that has given it the confidence that such litigation will likely be unsuccessful. Moreover, the Wine project is receiving legal counsel in case it is sued for infringement (LWN.net, 2005).

Many users just download and use the free Wine software. Before the firm entered the Apple market, knowledgeable Linux users were the “worst” competitors for the company. On the other hand, many of these users contribute to the project with testing and feedback, so the term “worst” was tongue-in-cheek.

7.2. Operational platform

7.2.1. Organizational capital

7.2.1.1. Capabilities, experiences and paths

CodeWeavers was founded in 1996, starting out as a programming consultancy firm. The mission of the company changed in early 1999 to one of helping ISVs to port their software from Windows to Linux. The company received venture capital in 2000. The business model in 2000 was to make use of the expected growth of the Linux desktop market.

One of CodeWeaver’s biggest customers during the Wine project period was Corel Corporation. Corel wanted to port its office products to Linux desktops; when the contract was withdrawn in 2001 (due to Corel’s sale), the firm decided to create a product out of the Wine project.

The company was “reborn” in 2002 as a product company with the launch of the CrossOver Office product. From 2002 to 2007, the company struggled with the economy and correspon-

dingly flat or minimal growth in sales. In 2007, a new opportunity arose when the Mac OS market opened up due to Apple's switch to the Intel processor architecture, thus creating great opportunities for the firm.

The founders and the management team are not fond of hybrid revenue models. They view their current, proprietary revenue model as unavoidable. However, the product’s success on the Mac OS market has required the company to adopt a traditional consumer-oriented revenue model using retail boxes. This business never took off for the Linux market. There are important differences between Linux and Mac users, from the firm’s experience. Where Linux users are more prone to “do-it-yourself,” Mac users tend to favor retail product with sufficient support. CodeWeavers has looked for this opportunity.

On the other hand, the most favored outcome for the firm would be to return to its original business mission, i.e. a service-oriented consultancy firm focusing on porting ISV software from Windows to Mac OS X and/or Linux.

CodeWeavers’ development staff is hired based on their programming proficiency from the Wine project. Even if most of the developers are fresh from college or universities, they often have more than adequate grades. The management team is also experienced in other fields. For instance, Jonathan Parshall is an accomplished historian, with an acclaimed book about the Battle of Midway (CodeWeavers, 2010j).

7.2.1.2. Intellectual property rights

Even if CodeWeavers does not retain the copyright on the Wine project, the company has significant influence since the maintainer of the project is part of the company’s management team. However, not every bit of the code produced by CodeWeavers will be let into the OSS project.47

As parts of the CrossOver product line, CodeWeavers has also developed tools for easier installation: an easy-to-understand GUI, virtualization software called “Bottles,” which helps organize the CrossOver installation, as well as other small scripts or programs that are proprietary.

CodeWeavers owns the corporate brand of CodeWeavers. However, the product brand crossover is not a registered brand, but rather a registered trademark.

7.2.1.3. Partner Agreements

CodeWeavers has no formal requirements for partner competence, but relies on the partner to deliver services if they are considered a VAR. Since all partners are essentially resellers, a standardized agreement is used in most cases, but in some cases, the company has signed non-disclosure agreements (NDAs).

7.2.2. Human capital

7.2.2.1. Community interaction

As stated before, CodeWeavers essentially works only with the Wine project. The company also hosts the project’s website, and has sponsored the annual conference WineConf. Even if CodeWeavers has a low-key presence on its website, WineConf 2008 is the best example of how the company sponsors the conference (WineHQ, 2010b).

---

47Alexandre Julliard has occasionally rejected code written by CodeWeavers for Wine. This is often due to that some may be more or less “dirty” workarounds to enable certain software to function, but it is not something that would benefit the OSS project.
CodeWeavers does not have a “customer community,” but paying customers can vote for applications or sign up as advocates for desired applications (CodeWeavers, 2010k).

### 7.2.2.2. Recruitment strategy

The Wine project is also important for the company in the area of recruitment. Most developers are hired directly from the project based on their accomplishments. When a new developer starts to make solid contributions, the project maintainer, Alexandre Julliard, notifies the management team (CodeWeavers, 2010l).

Most of the company’s developers reside outside of the United States, with the majority living and working in Europe. Therefore, most Wineconfs are located in Europe as well.

### 7.2.3. Financial capital

Historically, CodeWeavers has not been a very profitable company. Its goal for 2007 was to be able to pay all salaries, which it did. From a positive angle, the company generates revenue from a broad range of sources, and thus is not dependent on a certain entity or market. It is crucial for the company to gain additional volume. Fluctuation in exchange rates hurts CodeWeavers, not only because of sales abroad, but also due to salaries. Nevertheless, one of the priorities for the company is to be able to hire more developers from the Wine project.

However, the focus on the Apple computers market already looks to provide CodeWeavers with a far more stable income and a profit. Revenues for 2007, approximately $1.9 million, were divided into four income streams: consumers (50%), corporate customers (approximately 20%), professional services (circa 20%) and retail (10%). Retail products were observed to be the most rapidly expanding business segment. Despite the financial crash of 2008, the company has seen increased revenues each year with almost $3 million in 2010. In addition, the firm has been profitable since 2007 and been able to accumulate a cash buffer, despite increasing salaries and more employees.

### 7.3. Offering

#### 7.3.1. Offer models

All products and prices are displayed in US dollars and are purchasable through CodeWeavers’ website. Larger customers are offered a sales contract and adapted pricing. In addition, resellers offer the product in their areas and in local currency, while VARs offer local support.

CodeWeavers also uses a retail channel, mainly for its CrossOver Games and CrossOver Mac. The products are packaged into boxes by RedFrog and shipped through distributors to shops in the United States. The company intends to expand this offering to other regions as well.

#### 7.3.2. Products

CodeWeavers offers three different versions of its CrossOver product: CrossOver Games, CrossOver Linux, and CrossOver Mac. All products are derived from Wine software enhanced with proprietary additions, such as graphical user interfaces (GUI), scripts and tools, to enhance operation and installation.

Another part of the offering consists of support and services, such as guarantees that the supported applications listed will be functional, regardless of the status of Free Wine. To some degree, other support services are also included in the product pricing.
In addition to the products, CodeWeavers also offers consultancy services for Wine, end user support for specific applications, porting services (making a Windows application run in Wine) and software testing and evaluation. These services are offered at an hourly rate.

7.3.3. Distribution model

7.3.3.1. Linux

CodeWeavers’ Linux products are primarily distributed through Internet downloads. A media kit with a serial code can be made available for an additional cost. The company has a two-tier distribution model for Linux customers, working directly with resellers. However, in Germany and the United Kingdom a three-tier model with distributors has been established.

There are three regional distributors; GT Enterprises is the distributor for India, while Resolvo is the distributor for both Singapore and Asia. GT Enterprises and Resolvo also act as OEM partners, helping customers to port products from Windows to Linux. In the United States, Techalign acts as the main distributor.

7.3.3.2. Mac OS X

CrossOver Mac has a different type of distribution. In addition to the normal downloadable version, the Mac OS product is also available as a boxed retail version in the United States, which is distributed to retail channels (Amazon.com, 2010) by the main distributor Navarre and educational distributor Douglas Stewart. However, according to the company, changes in the retail market are an issue for the company to tackle.

The distributor NetJapan has an exclusive deal to distribute the product in the Japanese market, while Pica is responsible for the Australian market. Atlantia distributes the product in Canada.

7.3.4. Revenue model

CodeWeavers offers its products with a proprietary license, including support services. The license gives the licensee the right to upgrade the product for a set period. For the standard product, this is six months, while the professional and games versions cover a full year. The professional customer is also able to renew support for an additional year at a discount (CodeWeavers, 2010m).

In addition, other services are charged at an hourly rate. The pricing of the product is set to be competitive compared to its competitors or substitutions.

The relationship between different markets has undergone significant changes; for example, the retail or direct customer sales are now 50/50 between Linux and Mac OS. However, retail growth is almost entirely on the Mac OS side.

There have also been changes regarding corporate customers as well. Where the ratio of customers is 60/40 in favor of Linux, the services have gone from nearly 100% based on Linux to approximately 75% based on Mac.
8. Case 3: MySQL AB

MySQL AB is a Swedish firm founded in 1995. It is closely connected to the database product MySQL. The founders, David Axmark, Allan Larsson and Michael “Monty” Widenius, had earlier setups when they worked as consultants, but after their success with the database, the firm was molded around the product. The name MySQL is derived from Monty’s daughter My. The dolphin, which is part of the company logo, is named Sakila, which is a feminine SiSwati word.

MySQL gained popularity in the late 1990s as a part of the so-called “LAMP stack” (Linux, Apache, MySQL and PHP), which became a standard way to set up a Linux-based web server. The concept of the stack is still widely used, but nowadays PostgreSQL (making the “LAPP stack”) sometimes replaces the MySQL server.

Being popular is not always pleasant. MySQL has often been subjected to extensive criticism. Most issues regarding the functionality of the database, however, were remedied with the release of Version 5 in 2005. MySQL AB became the holding company after venture capital entered the firm. Until 2008, MySQL was privately-held with its headquarters in Uppsala, Sweden, and subsidiaries in Finland, France, Germany, Ireland, Japan, Russia and the USA. In 2008, Sun Microsystems acquired MySQL and in 2009 Oracle announced its intention to acquire Sun Microsystems, but the deal was not completed until 2010, due to questions about how Oracle would treat MySQL (O’Grady, 2009).\(^4^8\)

This presentation poses a problem, since MySQL is no longer is company. Therefore, the facts presented here are a mix between the former situation, up until 2008, and the present situation. A statement of what time period that is described will accompany each section.

8.1. Business position

This section will be a mix of the past, especially concerning business ideas, visions and goals, and the present.

8.1.1. Business idea

MySQL’s main business idea before 2008 was to provide a database to those who build online services and applications.

8.1.2. Vision and goals

The company understands that the world is going online; therefore, its vision is for a new type of data management. Regarding the culture of the firm, MySQL has made up a list of its values:

We want the MySQL server to be:
- The best and the most used database in the world
- Available and affordable for all
- Easy to use
- Continuously improved while remaining fast and safe
- Fun to use and improve
- Free from bugs

MySQL AB and the people of MySQL AB:

\(^4^8\) The reference is a summary of the discussions regarding the acquisition and implications of MySQL.
The company’s goal was to be the OSS database that would be “first in mind” for customers interested in OSS:

Our marketing goal is not to convince people about Open Source, but to make sure that those convinced about Open Source doesn’t have any other database in mind than MySQL. That is our goal. We are not trying to convert them from a classic attitude to the new one. There is no point, we let other people do that.\(^49\)

When asked about the strategy in five years (2010), the CEO had a clear picture:

Focus, focus, focus! Stick to thy last. The database market is worth 15 billion [dollar]. Of course we have to add functionality we don’t have today. On Monday, the fifth edition is released and it is a big, big step into new markets. We will add other things needed and that opens up enormous markets for us. In this we have our faith. We have our strategy, we have our roadmap and everything, now all we have to do is to work hard.\(^50\)

One of the clear visions for the company was to “commoditize” the database market through a “disruptive” business model, offering a highly functional database product with low prices and through this gain a large portion of that market.

### 8.1.3. Competitive strategies

The company’s main competitive strategy was to have a perfectly designed product for the online world, sold under a “disruptive” business model. Moreover, its product is OSS; it boasts a vast number of users, 70 000 downloads per day and at least 12 million installed databases in 2009 (Barrett, 2009) and an active network of developers. This kept development costs low for the company, enabling a low-cost, economy-of-scale strategy.

The company retained the copyright for the core product, which enabled a dual-licensing strategy, offering the product as OSS and as proprietary software to OEM customers. In addition, licensing the software as OSS in 1999 (a special MySQL license) created the LAMP stack for web servers, which was already widely used in the dot-com era (Dougherty, 2001).\(^51\) This contributed to the attention for and rapid propagation of the product. Moreover, it created a large ecosystem of tools and additions used with the product, which later were the subject of a conference (O’Reilly, 2010).

### 8.1.4. Branding and marketing strategy

In this section, the past strategy of the company will be mixed with the present, when the product is part of Oracle. The reason for this is to describe how the company marketed itself and its product, and to contrast the present situation.

#### 8.1.4.1. MySQL 2007

MySQL maintained one brand, MySQL, both as a corporate and product brand. Under Sun Microsystems, and under Oracle, the brand has become a product brand. What is unique is that the community project is also named MySQL.

---

\(^49\) Interview, Mårten Mickos, CEO, MySQL AB, October 28, 2005. The quote has been translated from Swedish.

\(^50\) Ibid.

\(^51\) The “LAMP stack” consisted of Linux as the operating system, Apache as the web server software, a MySQL as the database, and Perl, Python or PHP as the web programming language.
As a company, MySQL always had a very serious commitment to the OSS community. For example, the company had a vice president, Kaj Arnö, assigned to directly handle such contacts; he retained that position after Sun acquired MySQL. The co-founder and vice-president, David Axmark, was also highly involved with the community.

The branding strategy was initially low-key, mainly due to lack of funds. When Mr. Mickos entered as CEO and the firm attracted venture capitalists, the branding and marketing activities were increased. This renewed marketing focus, through PR and media interest, actually increased the number of product downloads, which also (as expected) attracted more customers and investors.

However, it also affected the community. Even if MySQL values the community interaction, it has had its share of mishaps and misunderstandings in its relationship with the community. Some business decisions can create many angry voices, such as asking users to register (which was a misunderstanding; a registration was only necessary on the commercial website). Regarding the business agreement with SCO, in the latter case the outcry seems to have been blown up by the media (Jones, 2005). Another disappointment had to do with the decision to remove the tar ball (a way to distribute binary code) for the community server and only make the enterprise version available for paying customers (Clarke, 2007).

Nevertheless, in 2005 the CEO illustrated how the brand worked with both customers and the community. He wrote a parable about how a popular nightclub handles VIPs and regular guests:

It’s like starting a nightclub or a discotheque. You have to give free tickets to the VIPs and so all the celebrities get in. They attract the normal folks and those people are willing to queue and pay, in order to get the access to the same things the celebrities don’t have to queue or pay for. But it works. It is the same situation for us, we have to let the open source community in for free and ahead of the queue, while our regular customers have to queue and then pay for it. But both groups are content.52

Even as MySQL is a company and has to be profitable in the end, it realizes the real value the community has for its business. The community is a part of the success of the company.

The brand was promoted through the OSS project and the website it maintains (which also promotes the product), as well as through different marketing campaigns, including indirect sales and through fairs and events.

8.1.4.2. MySQL and Sun Microsystems (2008 - 2009)

When Sun Microsystems acquired MySQL for $1 billion in 2008 (MySQL, 2008), there was some consternation inside Sun, since the company had actively contributed to and promoted rival PostgreSQL (e.g. Shah, 2008).

The real reason was not to gain access to the product, but to the knowledge. “The reason Sun bought MySQL was to learn how to make business on open source.”53 It was the hope that the MySQL brand, signifying a product, a community and a business model, would bring business back to Sun (Vance, 2009).

During the year as part of Sun, some of the key individuals left, such as co-founder “Monty” Widenius (Asay, 2009a), Mårten Mickos (Asay, 2009b), and David Axmark (Niccolai, 2008), while Kaj Arnö stayed throughout the Sun time, resigning from Oracle in September 2010 (Clarke, 2010b).

---

52 Interview, Mårten Mickos, CEO, MySQL AB, October 28, 2005. The quote has been translated from Swedish.
53 Phone interview, Kaj Arnö, Vice President Database Community, Sun Microsystems Inc, September 30, 2009.
8.1.4.3. MySQL and Oracle (2010 - present)

The financial crisis hit Sun Microsystems hard in the fall of 2008, and the company had to be restructured and was looking for buyers, but failed to see who had interest in purchasing the company (Kawamoto, 2008).

However, before Oracle acquired Sun Microsystems in 2009 (McDougall, 2009), Sun had rejected a formal offer from the competitor IBM (Kawamoto, 2009). Even if there is still speculation about why Oracle bought Sun, Oracle clearly had interest in Sun software technology, such as Java and Solaris (Lohr, 2009). However, MySQL turned out to be something else, since the EU was reluctant to ratify the whole deal (Lohr and Kanter, 2009). After months of negotiating, the EU finally relented in January 2010 (Saltmarsh, 2010).

8.1.5. Pricing strategy

The former and current pricing strategy is “low price”.


The company culture of MySQL was unique; since most of the development is made from home, there is of course a problem establishing some sort of culture for the developers. The Scandinavian management style, with openness and the right to dissent, was appreciated by most. This management style, really a mix between the open and consensus-driven Swedish leadership and the more straightforward Finnish “management by perkele”54 (e.g. Lämsä, 2010) created a unique culture, where drinking songs and vodka shots were a given.

The lack of central offices also made the annual developer meetings more important. The developers also shared a number of non-programming interests, the most common being photography, running and cooking. This supported the sense of physical community, because when they met they could also share a jog.

8.1.7. Value constellation mapping

There are two value constellations, one from 2007 and one from 2010. The 2007 version is based on interviews and secondary data collected before December 2007, and some secondary data from the Internet Archive has been added, from December 14, 2007 (Internet Archive, 2010b). The current version used the interview from 2008 and the product website in the fall 2010.

---

54 Perkele is a Finnish profanity, and the expression “management by perkele” is a Swedish expression to explain the differences in Swedish and Finnish management culture (see Pennanen, 2004).
8.1.8. Value constellation map 2007

Figure 11: Value constellation map of MySQL in 2007.

8.1.8.1. Community

The community served three purposes: *brand recognition, quality assurance* and *recruitment base*. MySQL was one of the most popular OSS in the world (MySQL, 2007b). The company was proud of being a part of the open-source community, which was constantly communicated. One sign of this close connection was the direct link to the community on the corporate website (Internet Archive, 2010i).

The firm had registered more than 11 million product implementations (MySQL, 2007c). The number of downloads was believed to be ten times or more than actual installations, making it clear that the number of downloads did not have to signify usage. This problem is the same for most OSS projects and products, which has been studied by several researchers (Lee, Kim and Gupta, 2009). However, every visit or downloaded copy increased the brand recognition and added potential customers or developers.

Even if most of community did not work with product development, it provided the company with brand recognition, valuable testing and quality assurance, which was a direct benefit for everyone involved.

At the time, most of MySQL’s developers were recruited directly from the community. In addition, in accordance with the OSS credo, MySQL used the actual contributions to review the competence and capacity of the applicants. One of the co-founders stated that he probably would not have been hired as a developer for MySQL if he had to compete with those hired during that year.55

The community contributed to two of the products, the *Enterprise* and *Embedded Database* product. If a developer wanted to contribute to the core database engine, he or she had to sign a *Contributor License Agreement* (CLA), which enabled MySQL to retain the intellectual property for the product (Internet Archive, 2010j).

55 Interview, David Axmark, Vice-President and Co-founder, MySQL AB, October 12, 2007.
8.1.8.2. Partners

MySQL had a wide array of partners, and the partner ecosystem served most customer needs. For such a small company, the company had a large number of partners. Some of the partners were connected to the OEM business, but there were also third party vendors who added functionality to the software suite (MySQL, 2007d).

In 2007, MySQL introduced a new business partner program, called MySQL Enterprise Connection Alliance (MECA), which was an extension of its earlier efforts (MySQL, 2007e). The partner ecosystem consisted of a large number of partners, divided into nine categories: ASP, consulting, distributors, hosting, ISV, OEM, platform, resellers and training. Some of the partners worked in more than three categories, but the majority was only connected to one or two (Internet Archive, 2010j).

In addition, the MECA program had three different partners levels, from the entry level “Ready”, through “Gold” to “Platinum”. The entry level was free of charge, while the intermediate “Gold” level cost $2999 to obtain. The highest level was by invitation only, and each case was handled individually (Internet Archive, 2010k).

This ecosystem was what allowed MySQL to offer solutions to a wide array of customers all over the world. It was also a major differentiating factor from other database vendors and part the company’s rapid success (Fegreus, 2003).

Some partners were among the world’s largest computer companies, for example Dell, HP and IBM, which integrated the product with their own products and offered training and consulting services (Internet Archive, 2010j).

The number of OEM partners who offered proprietary solutions with MySQL embedded was higher than the official 20, since a majority of these were not interested in publicly announcing that they worked with MySQL. However, these were considered very important for the company (Darrow, 2007).

8.1.8.3. Owners

Private investors, angel investors and venture capitalists owned MySQL (Internet Archive, 2010l). The company had significant investments from highly-acclaimed venture capitalists. There were also rumors about an initial public offering (Aslett, 2007).

8.1.8.4. Customers

Since the company was founded, the number of corporate customers has steadily risen. However, the number of paying customers was very low compared to the number of downloads. The aspect of “free-riders”, i.e. users that download the product and use it, without paying for licenses or services, is interesting. The company calculated that there was one paying customer per 1000 users. The reason for the rapid expansion was to gain more paying customers by adding more, non-paying users. For the CEO, this was not an issue, since all users contributed to spreading the product. In addition, with 100 million downloads, there were enough customers (Internet Archive, 2010a).

Moreover, even if most users were non-paying users, and some of those were developers, other paying customers were also part of the community, like Google, who made significant contributions to MySQL (Babcock, 2007).

Nevertheless, the widespread reputation in the OSS community also enabled MySQL to attract internationally acclaimed investors to infuse large sums into the company (Internet Archive, 2010m).

One of the problems, as seen from a marketing perspective, was that the OEM customers who embedded MySQL within their own products, the main business for MySQL, were not always interested in advertising this fact, despite these being important for the company (Darrow, 2007).
8.1.9. Value constellation map of MySQL 2010

The most important difference is that MySQL ceased to be a business entity when Sun Microsystems acquired the company in 2008. After Oracle acquired Sun, MySQL became one product in Oracle’s portfolio of products. Therefore, the three different offerings that came with MySQL are included as part of the product in Figure 12.

Figure 12: Value constellation map of MySQL in 2010.

8.1.9.1. Communities

There were several major changes to MySQL’s community contact after Oracle acquired Sun and MySQL. Two different projects, Drizzle and MariaDB, were forked from the MySQL codebase. Drizzle is a community-based project aiming towards cloud computing (Drizzle, 2010), while MariaDB is a direct competitor to MySQL (MariaDB, 2010). More important, one of the founders of MySQL, Michael “Monty” Widenius, formed a company and consortium with MariaDB called Open Database Alliance (Murray, 2009).

In addition, Oracle made small but significant changes to the MySQL website, among those removing the direct link to the community website and making the community versions available as downloads, without any connection to the community (MySQL, 2010e).

There are concerns that Oracle will try to slow the community contributions, based on earlier acquisitions like Innobase and Berkeley DB (Byfield, 2009). However, the community is still active, and at the time this thesis is written, it is too early to say anything about the consequences.

8.1.9.2. Partners

MySQL is included as a product in the Oracle PartnerNetwork (OPN) (Oracle, 2010b). This was something that Sun refrained from doing, instead keeping the existing MECA program (Glore, 2008). In comparison, the OPN is much larger than MECA; on the other hand, the former hosts a much wider array of products.

The number of MYSQL partners has decreased, which is likely explained by the fact that not all former partners have yet applied for membership in OPN. A basic search under Solu-
tions for MySQL shows only 21 hits, with one company occurring three times (Oracle, 2010c).

OPN is divided into five different levels, all with different obligations for the partner, from the entry level *Remarkeret* through *Silver, Gold, Platinum* and finally *Diamond*. The entry level is free, but the partner is managed by one of the *Value-Added Distributors* (VAD) who is responsible for the geographical area. In order to become a VAD, the partner needs to be a Platinum or Diamond level partner.

To attain the Silver and Gold levels, the partner has to pay a yearly fee, $500 for Silver and $2,500 for Gold. For the Platinum and Diamond levels, the fee is $9,995. However, the partner must attain a number of specializations through training and certifications, five for Platinum and twenty for Diamond, including five advanced specializations (Oracle, 2010d).

### 8.1.9.3. Owners

MySQL is fully-owned by Oracle Corporation.

### 8.1.9.4. Customers

MySQL customers can join the *Oracle Technology Network Community* (OTNC) to collaborate with Oracle about the products (Oracle, 2010e). The effect on MySQL customers after the acquisition was discussed in the media at the time of the announcement (King, 2009; Dziuba, 2009). However, after the deal was confirmed in January 2010, very little was written, with the exception of the Swedish magazine *Computer Sweden* (Sundkvist, 2010) where the image was one of uncertainty and doubt among the Swedish MySQL customers.

Moreover, every acquisition brings uncertainty among customers, and differences between the corporate cultures may worry them. The way information flows in one company may be different in another, which can also add to the worry. The lively debate and a competitive product from one of its co-founders did nothing to limit the confusion. The way in which Oracle handled earlier OSS acquisitions, as commented on in Byfield (2009) and Dziuba (2009), should be of concern; since there already exists forks of the product, however, the fears may be speculative.

### 8.1.10. Competitors

This section will only concern MySQL before the company was acquired by Sun Microsystems. All primary and secondary data was collected before December 2007. The section is therefore written in past tense.

MySQL identified three different kinds of competitors: the market leaders, firm-based OSS products and the OSS projects themselves. Among the market-leading databases, Oracle was the most vocal competitor. Oracle is the dominant world leader, having acquired numerous OSS companies. In 2005, Oracle bought the Finnish firm Innobase whose product, InnoDB, was distributed with MySQL (Oracle, 2005), and SleepyCat that produced Berkeley DB in 2006 (Oracle, 2006).

The premier commercial OSS database EnterpriseDB was directly aimed at MySQL, but based on the PostgreSQL database (Danielsson, 2007). The company positioned itself against Oracle, MySQL and PostgreSQL by offering specific solutions aimed towards those three (EnterpriseDB, 2007). One legal difference between the companies was that MySQL retained the original copyright for its product, while EnterpriseDB did not.

As for competing OSS projects, PostgreSQL is a popular and powerful OSS database, licensed under the BSD license. This license allows any user to make free use of the product, even repackage it and charge a proprietary license for it, making it a very widespread database engine. This is why EnterpriseDB is free to make a proprietary product based on the source code.
In the terms of actual users, PostgreSQL did not boast the same number of users as MySQL, but given the freedom of its license, it was very hard to track down its actual usage. In the European FLOSS report, the numbers were quite clear with 71% of the planned or installed base using MySQL, while PostgreSQL had 14% (Wichmann, 2002a). Even if there was other OSS database engine software, PostgreSQL and MySQL were by far the most popular.

8.2. Operational platform

The operational platform looks at the internal resources of a company. Therefore, the data presented below is dated before Sun acquired MySQL in 2008.

8.2.1. Organizational capital

8.2.1.1. Capabilities, experiences and paths

David Axmark and Michael “Monty” Widenius, with the help of the financier Allan Larsson, founded the company MySQL. It was in this firm that the database engine was initially developed. The founders realized the potential for the firm, so they decided to recruit an experienced managing director. When Mårten Mickos was asked, he initially turned the offer down. However, since he saw great potential in the MySQL opportunity in his role as advisor for the financing round, he finally accepted. With his assistance, MySQL managed to attract large venture capital investors, who also provided help with the company’s management. The management team and investors of MySQL were well experienced in the matters of the software and database industry and OSS.

The product always had a proprietary license attached to it, especially the early Windows versions. However, MySQL decided to license its product with GPL in 2000 (MySQL, 2001).\(^{56}\) In addition, the libraries used LGPL and MySQL was among the first firms to make use of a dual-licensing scheme. However, in 2006, MySQL decided to change licensing for its libraries from LGPL to GPL to stem misuse of the product. This led to unfortunate consequences, since it also affected other bundled OSS, which in turn forced MySQL to make a “FOSS exception” (Shankland, 2004a).

Even if the decision to make MySQL OSS was made relatively early, the firm has always focused on becoming profitable. This was the main reason for experimenting with different licensing schemes for its product.

The founders are regarded as competent programmers, and many of the developers are employed based on their achievements in the community. The company has been able to recruit many very competent software developers throughout the years. However, MySQL also realized that it was difficult to recruit OSS developers to work on the database engine since it is very complex; it is therefore quite hard for a developer, working on a hobby basis, to absorb the knowledge he or she needs to be able to develop it. Very few have managed this effort, and most have been hired to work for the company.

8.2.1.2. Intellectual property rights

In order to retain copyrights to pursue the dual-licensing scheme, a developer interested in working with the database engine had to sign a contract, the MySQL Contributor License Agreement, where his or her copyright for the code was signed over to the company.

---

\(^{56}\) The original press release was dated June 28, 2000. However, the press release was updated July 2001.
In addition to the database engine, MySQL also owns the IPR for MySQL Cluster, developed by Ericsson but spun off to Alzato, which MySQL acquired in 2003 (MySQL, 2003). MySQL owned the brand for MySQL, which was used as a corporate, product and community brand.

8.2.1.3. Partner Agreements

In 2007, MySQL introduced the MySQL Enterprise Connection Alliance (MECA) to create a consistent framework for its already sizeable partner network (MySQL, 2007e). The partner “ecosystem” was one of the most important parts of MySQL’s business (Fegreus, 2003).

8.2.2. Human capital

8.2.2.1. Community interaction

MySQL has always considered community interaction as vital for the company. The company compared itself to a nightclub, where the community developers resembled VIPs and were fundamental for the firm’s success. In 2005, Kaj Arnö became the official liaison between senior management and the community when he assumed the post of Vice President for Community Relations (MySQL, 2005).

However, MySQL was a company first and sometimes managerial decisions created friction in its interaction with the community, such as misunderstandings regarding registering with the commercial website, or the deal with SCO (Jones, 2005). Perhaps this was due to changes in the MySQL Community Server in 2007 (Arnö, 2007), to which many in the community reacted negatively (Clarke, 2007).

Nevertheless, the company values the responses and commentaries from the community, and even if financial necessities forced some of the decisions that caused the community’s ire, the company tried to ease the decisions through open communication and information.

8.2.2.2. Recruitment strategy

Even if direct contributions were not as important for MySQL as for other companies, the recruitment opportunity was. The key recruitment strategy for MySQL was to assess the competence and ability of applicant developers through their contributions to the community. Many of the community-recruited developers worked from home, and if there were a certain number of developers in the same country, a local subsidiary was founded, mostly for tax reasons. In 2007, the company had subsidiaries in eight countries (MySQL, 2008).

8.2.2.3. Financial capital

MySQL had not been profitable since 2001, but it had a positive cash flow. The company was positioned to improve its presence and brand name in the market, in addition to investing in development for product improvement. Moreover, rumors that the company was gearing up for an Initial Public Offering (IPO) existed before Sun Microsystem acquired the company (Aslett, 2007). Table 12 below summarizes the financial statements from 2004 to 2007.
### 8.2.3. Investors

MySQL was privately held, but attracted investors. The biggest owner was Benchmark Capital Partners IV, followed by Open Oceans, Index Ventures II, Scope Venture Capital I, Institutional Venture Partners XI, Holtron Capital Fund I, Crédit Suisse Life and Pensions, SAP, and Intel Capital Corporation (MySQL, 2008b).

### 8.3. Offering

The offering has been studied based on the how MySQL offered its products before being acquired by Sun, and how Oracle now offers MySQL after its acquisition.

#### 8.3.1. Offer models

Through the following avenues, the customer could choose from a number of ways to purchase the offerings: the MySQL Online Shop (web shop), the *MySQL Enterprise Sales Team*, the *MySQL OEM Sales Team*, the *MySQL Consulting Team*, the *Channel Partners* or a *General Service Administration* (GSA) Schedule (American governmental purchases) (Internet Archive, 2010n).

The only difference between the offerings is that the product is part of Oracle, with its online store, partner network and sales resources (MySQL, 2010g).

#### 8.3.2. Products

MySQL differentiated between its software products and its service products. In 2007, MySQL offered three different software products, *MySQL Enterprise*, *MySQL Cluster*, and *MySQL Embedded database* (Internet Archive, 2010o). Aside from the service included in its software products, MySQL also offered training and certification services, consulting services, and support services for its OEM-embedded database product (Internet Archive, 2010p).

The MySQL software and service products remain the same under Oracle (Oracle, 2010f).

#### 8.3.3. Distribution model

All MySQL products were and are distributed through downloads from the Internet.

#### 8.3.4. Revenue models (2007)

MySQL offered its software product with two revenue models, a subscription model with OSS licensing and services and support included, and a traditional proprietary license for OEM or embedded versions.
The subscriptions for MySQL Enterprise came in four different tiers: Basic, Silver, Gold, and Platinum, with prices ranging from $599 annually per server for the Basic, to $4,999 for the Platinum. The differences between the tiers were access to tools and levels of service. A number of options such as indemnification were open for gold and platinum customers, while account management was only available to platinum customers (Internet Archive, 2010q).

For the proprietary versions, no official price was published, but unofficially the price was set at around €400 (Bank of Finland, 2010). The pricing for certification courses varied between $1,495 and $2,495, and certification was offered in bundles, with a 12% to 25% discount, depending on the number of students (Internet Archive, 2010r). As for the other services, no official pricing was available.

8.3.5. Revenue models (2010)

The main changes for the revenue models are that Oracle has decreased the number of options, for instance removing the Basic and Silver price tiers (Asay, 2010b). Oracle offers MySQL in three editions, MySQL Standard Edition, MySQL Enterprise Edition, and MySQL Cluster Carrier Grade Edition. The price, per year and server varies from $2,000 for the Standard Edition to $5,000 for the Enterprise Edition to $10,000 for the Cluster Carrier Grade Edition (MySQL, 2010h). No official pricing exists for the OEM/Embedded versions.

As for the training and certification services, MySQL is part of the Oracle University. Pricing and availability of the courses differ between geographical areas (Oracle, 2010g).
9. Case 4: Red Hat Inc.

Red Hat was founded as ACC Corporation by Bob Young in 1994, selling UNIX and Linux accessories. In 1994, Marc Ewing created the first version of Red Hat Linux; Young acquired Ewing’s business, and after merging, the company was renamed Red Hat Software. The product was a distribution of Linux on a CD-ROM, together with a manual in a software box. The product was released on a six-month rolling schedule. Red Hat was the first OSS firm to make an IPO in 1999, and had a very successful introduction on the stock market, the eighth largest first-day gain in Wall Street history (Shankland, 1999). In addition, in 1999 Red Hat acquired Cygnus Solutions, the first open-source company, becoming the largest OSS company in the world (Red Hat, 2010a).

Red Hat management realized that the original business concept, selling retail products distributed via media in boxes, was deteriorating quickly as faster Internet connections made large downloads easier for everyone. The company decided to change its focus from retail customers towards business customers. Red Hat had installations at large corporations, but not significant, business-critical systems. One problem identified was the fast prototyping cycle, which made it impossible to certify the Linux products, i.e. ensure the quality, robustness and functioning of the system on large enterprise systems. Another problem was the lack of certification of system interoperability, since there was no defined hardware and software ecosystem. In order to attract larger enterprises these problems had to be addressed.

In 2003, Red Hat remade its business and launched Red Hat Enterprise Linux, a product with a longer lifecycle. In the same year, Red Hat initiated the Fedora project, which was aimed to be the community-driven Linux distribution avenue with shorter release cycles (Red Hat, 2010a; Fedora Project, 2010).

In 2006, Red Hat acquired the middleware company JBoss, widening its product portfolio away from operating systems only (Red Hat, 2006; LaMonica, 2006b). In the race for virtualization, Red Hat acquired Qumranet in 2008, adding functionality to its desktop product (Red Hat, 2008, Gardner, 2008).

9.1. Business position

9.1.1. Business idea

Red Hat considers itself to be the world’s leading open-source and Linux vendor. The main business concept has always been to aggregate products based on OSS. The company offers two different product ranges, both based on paid and unpaid code contributions.

The company firmly believes that OSS provides the most innovative, competitive, and sustainable basis for selling those subscriptions.

The business mission statement supports that contention. In 2007, the mission statement was formulated as follows:

In terms of the mission statement, we do have a mission statement that something likes something: “To build some quality product at the lowest possible cost supported by an active community of users and serviced by brand”. It puts together the concept of high value and low cost, community and brand.\(^57\)

The official mission statement in 2010 is somewhat different, and perhaps more of a vision:

---

\(^{57}\) Interview, Michael Tiemann, Vice President Open Source Affairs, Red Hat Inc., December 10, 2007.
“To be the catalyst in communities of customers, contributors, and partners creating better technology the open source way” (Red Hat, 2010b).

9.1.2. Vision and goals

The mission statement above is more in the line of a vision, largely due to its forward-looking content. No other company-encompassing vision statement can be found.

Goals are set primarily for the Sales department. “Sales is the lifeblood of our growth strategy,” according to Michael Tiemann. In order to develop its marketing strategies, goals are set for its diverse types of sales, like inside direct sales to the third-party reseller channel. The goals may look internally competitive, for example horizontal versus vertical and geographical versus centralized, but given the diversity of the sales systems this is to be expected.

From a marketing standpoint, the strategy has been to be disciplined when it comes to messaging, for example through its participation in press releases. The main goal is to develop instead of “consume” the brand.

9.1.3. Competitive strategies

Red Hat is proud to tout its OSS heritage and maintains that the performance and cost effectiveness of its enterprise solution based on OSS creates the most value for its customers (Red Hat, 2007d). While Red Hat’s enterprise Linux products are sold through subscriptions, the company still maintains a free beta testing Linux distribution called Fedora. Fedora is used as a prototyping and development platform, where Red Hat can test technology strategies for future generations of the enterprise product range (Fedora Project, 2010).

Even if Red Hat is considered highly priced compared to other Linux distributions (Prickett Morgan, 2010), it does have “cost effectiveness” as one of its leading slogans. Comparing the Red Hat enterprise product with other comparable products, such as UNIX and Windows Server, Red Hat is no longer in the lower price range. Looking at the total offering with with its associated services, but without a software license included, Red Hat does have a low cost alternative to that kind of competition (e.g. Cybersource, 2004).

In reality, Red Hat is aiming for is a differentiation strategy. Close to identical Linux distributions, such as CentOS and Oracle’s Unbreakable Linux, have not significantly hurt Red Hat (Asay, 2009c). However, in the application server market, where Red Hat offers the JBoss solutions, it is priced lower compared to the big commercial competitors such as IBM, Sun and BEA Systems.

9.1.4. Branding and marketing strategy

Red Hat is one of the most popular Linux distributions, and has been so for a number of years. Even if the company has geared itself towards larger enterprise customers, the brand recognition within the OSS community remains very high. Branding is also prioritized in its management, with a Senior Vice President for “people and brands”, which could signal the closeness of human resources and branding in an OSS company (Red Hat, 2010c).

58 Ibid.
59 Even if the actual product is sold through a license look-alike “subscription,” all the individual files remain open and downloadable from a vast array of servers worldwide. What the subscription covers, which is not included in the files, is support and services, a patch and upgrade system, and installation scripts which makes the implementation of the product easier. (or … which facilitates product implementation)
The company does not have a separate branding strategy between the community and its enterprise customers. Red Hat branding takes more of a unified and holistic standpoint, where some parts appeal more to the community. Michael Tiemann states: 

We see ourselves as innovators, we see ourselves as Open source true believers. We see ourselves as committed to the community in terms of participation. We are not just cheerleaders; we have people in the field actually taking the hard tackles.60

The participation is not only through paid development in OSS projects, like the Fedora Linux distribution, but also by actively working in other organizations like the Open Source Initiative (OSI). In the management team, Michael Tiemann serves as the community liaison. By giving Fedora its own brand, Red Hat wanted to communicate that it is an independent distribution, but supported by the company.

Red Hat’s great reputation within the OSS community may have been one of the factors for its successful public offering in 1999, which generated great interest from large investors. After the successful IPO in 1999 and the general hype of Linux, Red Hat quickly became a widely recognized brand. This also made Red Hat Linux the market leader among Linux distributions. Other commercial distributions like Caldera and TurboLinux also made public offerings, but never gained the interest that Red Hat did. Red Hat’s OSS project was also the largest Linux distribution with a market share close to 50 percent (Shankland, 2000).

The concept of creating an enterprise version of Linux, which Red Hat was the first to do, shows the innovative commitment of Red Hat and its targeting of key customers. In addition, the products have been ranked “top enterprise software” for several years, further establishing the corporate brand.

Red Hat also acquired JBoss, one of the best-known application servers (LaMonica, 2006b). As a close partner of JBoss, Red Hat had similar offerings, with subscriptions such as the preferred revenue model. With this acquisition, Red Hat implemented a forward vertical integration in order to strengthen the OSS platform it offered.

One important part of the corporate branding has been partnerships with multinational companies in the IT industry such as IBM, HP and Dell.

Red Hat’s brand and its products are promoted through direct sales, indirect sales, channels sales, and partnerships (such as pre-loaded computers), but mostly through the company website. It has also been a distinct policy for Red Hat to do virtually no advertising.

In its promotions, Red Hat drives a theme of focusing not on products but on what customers can do with the current level of technology. The website and online Red Hat Magazine do not promote a product in particular, but rather are aligned towards customer productivity, presenting success stories to enhance its claims.

The company also attends and sponsors selected fairs and events. However, Red Hat decided to drop the LinuxWorld conferences and other industry events in the fall of 2007, in order to focus more on its own events.

9.1.5. Pricing strategy

The product is not a low-price product (Pricket Morgan, 2010); rather, the pricing strategy is based on differentiation.

9.1.6. Culture

Red Hat strongly emphasizes that it stands for OSS ideals. With phrases like “We won’t compromise our values: courage, freedom, commitment, and accountability,” and “We also de-

60 Interview, Michael Tiemann, Vice President Open Source Affairs, Red Hat Inc., December 10, 2007
pend on the three ideals that represent our greatest strengths: truth, collaboration, and value,” the company aims to reinforce its idealistic corporate identity and culture (Red Hat 2010j).

As in many other corporations, the marketing department is in charge of maintaining the culture. However, the company tries to reinforce a culture of openness despite its relatively large size, threats from competitors and demands from investors. The culture and attitude may be a main reason for attracting the best developers from the OSS community. Allowing the developers to work with a variety of OSS project reinforces the attractiveness for developers. However, a major company needs more competences than developers; the challenge for Red Hat may be to create a culture for non-developers as well.

Nevertheless, further challenges lie in the transcendence of OSS that goes beyond corporate boundaries. In a large corporation, different aspects of the inside and outside worlds collide. Even if the company culture may be influenced, the OSS culture is harder to affect.

One result of its commitment to OSS and its persistence in its values is the fact that Red Hat acquired firms with proprietary software and released the software as open-source (as part of the company’s commercial offerings). Even if the acquisitions were somewhat erratic before 2002, later acquisitions have been more focused (Asay, 2009d).

9.1.7. Value constellation mapping

![Value mapping of Red Hat](image)

Figure 13: Value mapping of Red Hat.

9.1.7.1. Communities

Red Hat is one of the largest contributors to OSS and its developers are active in hundreds of projects (Sundaram, 2010), of which Red Hat leads about 40 (Red Hat, 2010j).

However, this was not always the case. When Red Hat decided to change the development cycle of its Linux distribution, it defined a release process that effectively shut out customers and community developers. The new product was to be more closed in order to control its content and prolong its lifecycle. As a result, the important external users and developers were effectively shut out.
The company was on the verge of losing one of its most important assets, the open innovation (and bug testing) from customers and users. In order to avoid this, it created the Fedora distribution, which is unsupported, but has faster cycles, making it easier for developers to test new concepts that may be included in the commercial product. The enterprise edition is now continuously improved by the Fedora distribution.

Red Hat contributes to many projects with paid developers. One of the most important concepts of OSS is that independent developers and users contribute to the projects. It is important to realize that both users and developers test and solve problems that matter to them, but which are not necessarily the same problems the company wants addressed. Developers are very difficult to manage using traditional methods, but their contributions, when aggregated across hundreds of thousands of customers and millions of open-source developers, provide an important competitive advantage over proprietary competitors. Therefore, Red Hat has allocated resources to facilitate its communities, such as dedicated websites (Opensource.com, 2010; Fedora Project, 2010, etc.) with information and communication tools.

After acquiring JBoss in 2006, Red Hat now hosts two main community projects, the Fedora project (Fedora Project, 2010) and the JBoss community (JBoss, 2010). As previously described in the Fedora discussion, even if Red Hat is the main contributor, other contributions from the OSS projects must be funneled to the enterprise products in a controlled manner. Most of the other OSS projects with which Red Hat is involved are mostly connected to the Enterprise Linux product (e.g. Red Hat, 2010k).

9.1.7.2. Partners

Red Hat has an extensive reseller program, with different tracks depending on the products, such as infrastructure and virtualization, middleware or training (see Figure 13). The partner program has three levels, Ready, Advanced Business Partner, and Premier Business Partner, with requirements and benefits for each level. These three levels include not only the resellers, but also the partner categories Software Vendors and System Integrators (see below) (Red Hat, 2010l). Figure 14 below illustrates these partner levels.

![Figure 14: Red Hat Partner Levels (Red Hat, 2010m).](image)

In general, the reseller partner program requires the partner to submit to both training and sales commitments (in the business plan). Note that training partners are required to provide adequate courses, which requires higher competence and commitment for the partner.
Red Hat has a number of partner categories that provide value-added services or third-party products, such as hardware and software. These are organized into seven different groups: Alliance, Independent hardware Vendors (IHV), Hosting Partners, OEM Partners, Runtime Partners, System Integrators, and ISVs.

The Alliance partner level, where Red Hat collaborates with hardware CPU platform providers Intel and AMD, is a part of optimizing the product’s performance on these architectures (Red Hat, 2010n).

The Hardware vendor partners provide hardware certified for Red Hat products. The program is basically a way for a hardware vendor to market, for US$5,000 and some testing, its hardware products as certified to run a certain version of Red Hat Enterprise Linux. Some of the partners in this category are Dell, Fujitsu, HP and IBM (Red Hat, 2010).

Although the number of hardware partners is few, there are over a thousand ISVs with over three thousand applications certified for Red Hat Enterprise Linux or Enterprise Middleware (Red Hat, 2010p).

Hosting partners are those that provide cloud or SaaS services based on Red Hat products. This category has two levels, Red Hat Advanced Hosting Partner and Red Hat Premier Hosting Partner, where the latter is the more advanced (Red Hat, 2010q). Even though the number of partners was below ten, this is a segment that would later increase (Shankland, 2009).

The OEM partners are the seven largest hardware and software vendors to which Red Hat provides platform collaboration. Each of the seven partners has a separate section detailing the cooperation between Red Hat and the partner company (Red Hat, 2010r).

Red Hat’s runtime partner program is aimed towards companies offering appliances, i.e. products that bundle hardware and software in one package, and is a combination of an IHV and an ISV. The partner pays a one-time fee for each appliance, receives support for the entire lifecycle of the device (Red Hat, 2010s).

The final partner category is called System Integrators. Red Hat identifies two global partners in this category, HP and Infosys, where each partner provides different solutions (Red Hat, 2010t).

9.1.7.3. Owners

Red Hat is a publicly held company with a mutual fund as the top owner (Red Hat, 2010u). The eventual influence these kinds of shareholders have on a company like Red Hat is of course considered inside information. The relatively few active owners in privately held companies have a significantly larger impact on a firm’s strategy, much more so than they might in publicly held companies where the owners may be more passive. In this particular case, where Red Hat is the only publicly held company, the difference is of course even more significant.

The ownership structure of Red Hat is therefore of little interest for this research, and did not influence the outcome of this thesis.

9.1.7.4. Customers

Red Hat has many customers in different industries. The success stories on its corporate website number more than 200 in 12 different industries (Red Hat, 2010v). The majority of the success stories relate to the Consumer Industry, but it is well known that Red Hat has a strong presence in the finance and government sectors.

With its present platform of products and services, the company is addressing a very horizontal, wide array of customers. In 2010, the company had also positioned itself well in the virtualization and cloud markets (Deagon, 2010).
9.1.7.5. Competitors

Even back in 2007, Red Hat envisioned Google as a threat. Although the companies do not compete in the same markets, Google products like ChromeOS are getting closer to Red Hat’s products and might affect future business (Wagner, 2009).

What I see as the future competition is Red Hat and Google. Google is an OSS company that sells proprietary services. Red Hat is an OSS company that sells open services. [...] We are in a world now where Open source have actually become the endgame. Google have said, “The way we will play the endgame is to exploit Open source technologies and build proprietary services that run on our massive infrastructure. [...] Then there is the Red Hat play, which is “we are going to build a platform that the customer can maintain”.61

Three years ago, Red Hat identified Sun Microsystems as a serious threat. The Solaris UNIX operating system has the most common operating system for industries with high demands on computing. Most of the UNIX operating systems were perceived as threats to Red Hat, but the main vendors like IBM and HP have downplayed their own UNIX brands and work with Red Hat, which made Sun Microsystems the only real remaining UNIX threat.

When Sun announced that it was going to make an OSS version of Solaris called OpenSolaris, it was seen as a significant threat to Red Hat (Shankland, 2005). However, since Oracle has acquired Sun, the situation may change. Oracle already competes with Red Hat with a fork of Red Hat Enterprise Linux, but seems to have had minimal adverse effect on Red Hat’s business (Asay, 2009c). Oracle has announced that its Linux product will be available for Sparc systems, in order to compete with HP and IBM. How this will affect Red Hat is still uncertain (Niccolai, 2010).

Among the Linux vendors, only Novell, which purchased SUSE in 2003, was positioned as a competitor to Red Hat. SUSE Linux was the second-largest Linux distribution, with a strong grip on the European market, mainly in Germany, before it was purchased. Novell offers similar products to Red Hat, but the product is complemented with proprietary products from Novell’s earlier product portfolio (Novell, 2010a). However, in November 2010, Novell was sold to Attachmate, and its product line was split into two groups, separating the Linux business from the other products (Shankland, 2010). In the short run, this will probably have a positive effect on Red Hat, but it is uncertain what will happen in the end.

Another strong competitor is Microsoft, one the largest software vendors in the world with strong positions on the operating systems market (IDC, 2010).62 Microsoft has traditionally been very hostile towards Linux, but surprised many when it signed an agreement with Novell wherein the products were harmonized, with Novell customers protected against lawsuits and Microsoft selling subscriptions for Novell’s products (Evers and Shankland, 2006).

In the application server or middleware market, Red Hat’s premier competitors are some of their closest partners, such as IBM. IBM has other offerings in several other markets, so while one division is a good partner in one market, another can be a fierce competitor. However, the biggest specialized competitor was BEA Systems and its Weblogic Solutions, which was the market leader for Java-based application servers. In 2008, Oracle acquired BEA and is now a main competitor for both of Red Hat’s businesses (Oracle, 2008).

Furthermore, Microsoft is also a competitor with its .NET products. However, the Microsoft product can only be used on Microsoft operating systems, while most others can be used on several hardware and software platforms. Moreover, Novell offers a product for the project

---

61 Interview, Michael Tiemann, Vice President Open Source Affairs, Red Hat Inc., December 10, 2007
62 However, the actual market share of an operating system is hard to measure since it only counts sold subscriptions and not those without subscriptions, so it could be misleading (Masnick, 2010). However, since this affects Red Hat’s commercial product, the statistics are valid for this purpose.
Mono, which is a platform-independent .NET implementation, which runs on Linux (Novell, 2010b).

9.2. Operational platform

9.2.1. Organizational capital

9.2.1.1. Capabilities, experiences and paths

Red Hat was founded in 1995 and was one of the first major Linux vendors, offering its “flavor” of Linux in a retail version via CD-ROM with manuals. In addition, Red Hat also offered consultancy and support services, but primarily on the American market (Red Hat, 2010a).

However, profits were low and the demand for Red Hat to become more profitable was sharpened after the IPO in 1999 and the so-called “dot-com bust” in 2001. In response, Red Hat introduced its Enterprise line of products in 2003, which targeted the enterprise rather than retail market. This change was successful, as Red Hat has proved since then (Red Hat, 2011a).

In 2006, Red Hat widened its product line by purchasing JBoss, one of the leading OSS middleware companies. JBoss develops a middleware software platform that handles web-enabled applications (which the customers program), and is for applications using the Service-Oriented Architecture (SOA) (Red Hat, 2006).

Red Hat has been the commercial market leader for several years, and when the company made the enterprise fork, the relationship between the company and the outside world suffered greatly. This included the relationship with its paying customers:

After we released the enterprise Linux, version three we started to get some very consistent feedback from our biggest customers saying, “You guys are more difficult to work with, in terms of new features, than even Microsoft. How the hell did this happen?”63

After this, Red Hat fully understood the importance of maintaining a balanced relationship with the ongoing community. The company has worked hard to rebuild the trust between the company, the community and its customers.

In essence, Red Hat is also the company that introduced the subscription model for OSS. Some have criticized this revenue model, mostly regarding the content of the product, while others have been more positive (Shankland, 2004b; Asay, 2006; 2007).

Red Hat was among the pioneers when it came to commercializing Linux distributions, but was also an early adapter for the new revenue models. In the beginning, a great part of the revenue came from selling accessories, like T-shirts and mugs. Red Hat has offered services for years, and not just consultancy services, but also training and certification of customers; in fact, Red Hat certification for Linux administrators is the oldest, still active, certification program for Linux (GoCertify, 2011).

The step from the retail business to the enterprise market should be considered a quantum leap for the industry. The question remains if any other Linux company made that step and still survives. The successful IPO and the company’s market lead were probably the reasons it managed to endure after all. In addition, Red Hat is among the few profitable companies that still adhere to the OSS principle. The company has been an innovator, not only considering its products, but also in how it makes revenue.

In 2007, Red Hat appointed a new CEO to better guide the growing corporation (Shankland, 2007). As a public company, the management team and board of directors must be expe-

---

63 Interview, Michael Tiemann, Vice President Open Source Affairs, Red Hat Inc., December 10, 2007
rienced and competent (Red Hat, 2011b). Since the company is also able to acquire inventive firms, many of the most competent developers and managers are still within the company. One example is Michael Tiemann, co-founder of the first all-OSS company Cygnus Solutions, who works closely with the OSS community as Vice President for Open Source Affairs at Red Hat and President for OSI (OSI, 2011).

9.2.1.2. Intellectual property rights

Red Hat does not have copyrights or other patents on its Linux core, but it does have a portfolio of other copyrights and patents. Red Hat is one of the co-founders of the Open Invention Network (OIN) (OIN, 2005), which is a mutual patent pool for companies to share their patents in a way that promotes innovation (OIN, 2011). Red Hat retains patents and copyrights for defensive purposes, but in addition, the company has published a policy and patent promise, which protects those who use them in reciprocal licenses (like GPL) (Red Hat, 2011c). However, Red Hat does not extend the promise to “light” licenses like BSD, because the code and patents may be picked up by hostile entities, which then could use the patent against the company.

9.2.2. Human capital

9.2.2.1. Community interaction

Red Hat emphasizes that it has also been a market leader in the Linux community, and as such, has been able to recruit a great number of top developers over the years (Red Hat, 2010g).

As presented in the value constellation map above, Red Hat hosts several websites assigned to OSS, both for projects like Fedora and JBoss, but also others. The interaction between the company, its customers and communities is important for the company, since it promotes collaboration, which is vital for the company (Red Hat, 2011d).

9.2.2.2. Recruitment strategy

Red Hat is a global corporation, with offices around the world. Even if developers are drawn to Red Hat through community projects, there is no official recruitment strategy. All available positions are posted on the company website (Red Hat 2011e).

9.2.3. Financial capital

Red Hat is a publicly-held company and has been profitable since 2004. It is traded on the New York Stock Exchange (NYSE), and accordingly all financial reports are public. Table 13 below shows financial results between 2007 and 2010 (Red Hat’s financial year ends in February).
**Table 13: Financial statements 2007 - 2010 (Red Hat, 2011f).**

<table>
<thead>
<tr>
<th>Summary, financial statements 2008 – 2010 (Red Hat Inc.)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Financial period:</strong></td>
</tr>
<tr>
<td><strong>Employees:</strong></td>
</tr>
<tr>
<td><strong>Revenue (KUS$):</strong></td>
</tr>
<tr>
<td><strong>Net Income (KUS$):</strong></td>
</tr>
<tr>
<td><strong>Total Assets (KUS$):</strong></td>
</tr>
</tbody>
</table>

### 9.2.4. Investors

In 1998, Intel, Netscape and the venture firm Greylock and Benchmark Partners invested in Red Hat. The investment created some commotion, since Intel took an interest in Linux (Johnson, 1998).

Since 1999, Red Hat has been publicly traded and the top current investors can be found on the company website (Red Hat, 2010u).

### 9.3. Offering

#### 9.3.1. Offer models

Red Hat sells its products both directly from the company website and indirectly through partners. The reseller channel is largely promoted through regulated discounts based on the partner category and commitment to Red Hat. Depending on the type of partner, some are entitled their own price lists, which are regulated through their respective agreements with Red Hat.

#### 9.3.1.1. Products

Red Hat offers a number of products directly from its website. However, different partners may offer the same products as part of a larger offering. In addition, Red Hat offers customized products with adapted content to the customer. These offerings are not made public and are difficult to evaluate. The products presented here are those Red Hat publicly offers in its online store (Red Hat, 2011g). Red Hat offers four product categories: *server operating systems, desktop operating systems, enterprise virtualization*, and *training*. All products, aside from the training products, are offered as subscriptions that are valid for one year.

#### 9.3.1.2. Server operating systems

The server operating system, *Red Hat Enterprise Linux*, is the main product line for Red Hat. The product is offered for three platforms: 32/64 bit x86 servers (produced by Intel and AMD), POWER servers (produced by IBM), and the zSeries (IBM mainframe). The product is offered based on the number of physical CPU slots as well as the number of guest operating systems (for virtualization). In addition, there are three different offerings: *self-supported, standard* and *premium*. The difference between the offerings depends on the level of support (Red Hat, 2011h).

#### 9.3.1.3. Desktop operating systems

Red Hat offers a desktop operating system, *Red Hat Enterprise Linux for Desktops or Workstations*, aimed at companies requiring managed computers with limited administrative possibilities for the user. The desktop version is directed towards the normal user, while the workstation version is for more advanced users (Red Hat, 2011i).
9.3.1.4. Enterprise virtualization

In addition to its operating system, Red Hat offers the Red Hat Enterprise Virtualization for Servers Starter Kit and Red Hat Enterprise Virtualization for Desktops, which allows for server virtualization (different operating systems running on the same machine) and thin clients (the system runs on a server, rather than on a desktop machine) (Red Hat, 2011j).

9.3.1.5. Training

Red Hat offers vouchers for Red Hat-based training, since partners hold most of the certification courses. Red Hat also offers online courses that prepare customers for the certification tests, which must be done in person (Red Hat, 2011k).

9.3.2. Distribution model

All Red Hat products are distributed electronically through download, the preferred channel of distribution. Other partners may offer pre-loaded software as a bundled offering. Red Hat also offers media kits for those who prefer that kind of distribution.

9.3.3. Revenue model

Red Hat offers its products and solutions through subscriptions. The subscription contains the software, upgrade of the software (through the Internet) and support connected to the software. The subscriptions offered through the website are valid for one year, and it is possible to sign longer agreements, but only through direct contact with a reseller or the sales department (Red Hat, 2011l). Subscriptions constitute the main income for the company (Red Hat, 2011f).

Red Hat also offers additional services such as Service Level Agreements (SLA), training and certification, as well as essential accessories, all through the company website. However, there have been some discussions regarding the Red Hat subscription model, claiming it limits freedoms normally associated with OSS. For example, a Red Hat customer is not able to have any other Linux distribution connected to the subscription (Murdock, 2004). Regardless of whether the subscription is proprietary, or to what degree, in this thesis it is considered a hybrid license.\[64\]

In 2007, Red Hat launched an initiative called Red Hat Exchange. The Red Hat Exchange offered a unified Service Level Agreement, enabling a customer to pick and choose services by any constellation of different OSS products. One of the companies who joined was MySQL (Hamm, 2007). However, interest was low, and since Red Hat’s ecosystem has been successful, the Red Hat Exchange has been discontinued (Kerner, 2010).

\[64\] Whether the subscription matches the open-sourced code, or limits the freedom of the user is not really a matter of importance unless a third party feels wronged and sues Red Hat. This issue may very well be discussed back and forth for years to come, as some are for and others against it (e.g. Asay, 2006; Asay, 2007; Murdock, 2006).
10. Analysis

This analysis section is comprised of five sections. The first three present constructs based on the three factors in the business model framework, the business position, operational platforms and offerings. These constructs are used to facilitate the analysis of the cases in the cross-case analysis (section 4). The final section covers the generic business model configurations found.

10.1. Business position analysis

After studying the four cases, it was clear that one aspect influenced the business position significantly, the way the companies interact with communities. In addition, other researchers have stated that communities are important for OSS vendors’ business models (Perr et al., 2010; Dahlander and Magnusson, 2005; von Krogh et al., 2003). Therefore, the strategies for interacting with the community are the key to understand how the companies have configured their business models and the analysis will focus on these.

In fact, after studying the cases, three distinctive constructs have been found. The constructs describe strategies for utilizing communities by OSS vendors, two with different focuses and a hybrid strategy combining the two. The two unique strategies are called the symbolic link and the product affinity. The names have been chosen to reflect the effects of the strategies; other possible terms include market-oriented and product-oriented.

10.1.1. Symbolic link strategy

Companies employing the symbolic link strategy mainly utilize communities as part of a marketing strategy, where the community connection creates goodwill for the corporate brand. The community link works in two ways: first, promoting openness and collaboration, which in turn fosters innovation; and second, providing a free platform for CSR.

The community provides publicity, spreads brand recognition and attracts paying customers. In addition, the numbers of users who download and use the products train themselves in using the software, thus positively affecting future employers or start-ups.

The companies able to use communities as symbolic links must be large enough to sponsor OSS projects or provide paid developers to make contributions. There are different levels of kinship to OSS, from contributor to all-OSS.

A contributor is defined as a company that sponsors a project, contributes with paid developers and distributes selected OSS from its websites. However, these companies gain revenue from sources other than OSS, often through combinations of hardware, services or proprietary software.

An all-OSS company is defined here as one working exclusively with OSS and OSS-related products. Such companies gain the majority of their revenues from OSS or services related to OSS.

The level of community association determines how effective the symbolic link is. A company with an all-OSS association is likely to see immediate positive effects, including brand recognition and a recruitment base for developers. A company with a less distinctive link will still benefit, but be less effective in this area.

In addition, depending on the customer category, some companies may decide to distance themselves from the community, since too close a relationship here may hurt customer relationships. However, this threat has decreased as OSS has become more common.
10.1.2. Product affinity strategy

Companies employing the product affinity strategy are dependent on community contributions for their products. The price and/or the performance of the offering are directly connected to normally one, but sometimes several OSS projects. These companies are usually small to medium-sized, and need the extra development resources the community provide.

The role of the community can vary, between providing part of the company’s bundled offering with beta testing and general quality assurance. No matter how the community are utilized, it is important that the company be considered legitimate and feed back contributions or in other ways support the community. These companies often openly communicate their support to the OSS community in general, but also their affinity to specific OSS projects.

The dependency of the community on companies employing the product affinity strategy directly affects the companies’ operational platforms and offerings. This is especially important for companies that bundle OSS in their product offerings.

10.1.3. Hybrid business strategy

The symbolic link and product affinity strategies can be considered opposites, with other strategies lying between these two extremes. These hybrid strategies employ a combination of the two strategies. A hybrid strategy may entail the brand recognition factor, but also the need for contributions from community developers, or the extra resources for quality assurance.

However, this position may be precarious, since the company needs to balance the demands of the customers and the will of the community. Should the company institutes changes that annoy the community the community may create a fork, which detracts developers from the main project. In addition, the evolution of the community project may be problematic for the company, if the direction is not the desired one. Both forks and undesirable evolutions of community projects affect the capabilities of the company in a negative way, which may hurt its ability to deliver commercial offerings.

One interesting hypothesis, not tested in this thesis, is to ascertain whether there is an evolution in these strategies, where an all-OSS company moves from product affinity, via a hybrid strategy, to a symbolic link strategy. Some data indicate this, but there is no further evidence to corroborate this shift.

10.2. Operational platform analysis

For companies employing a product affinity strategy, the operational platform is important, since the company is dependent on the community as a resource (and thus part of the operational platform). What matters is how the company is able to control or affect its community contributions.

Three construct describes how communities are used in two different platform patterns, concerning development: company-driven and community-controlled. In addition, there is a hybrid setup consisting of a mix of the two.

10.2.1. Company-driven platform

Companies with a company-driven development platform are in control of the output of a community. This can be achieved in many ways, either by hiring the majority of the active developers in an OSS project, or by supplying all contributions in a project, more or less taking the project over – with the consent of the current project maintainer. However, if the maintainer does not consent to a project takeover, there is a risk – at least for smaller projects
– that the company will fork the project. Such a fork may become a liability for the company, especially if the rest of the community sides with the original maintainer. This may result in a depreciation of the company’s goodwill from working with OSS.

For companies employing a company-driven platform, the community provides a base for recruitment, with opportunities to hire talented developers based on the quality of their code contributions.

The downside of using this platform is that most of the resources come from the company itself, providing salaries for the developers. On the other hand, the company can employ project management methods to manage the development.

### 10.2.2. Community-controlled platform

For small companies or companies working with large communities, the only option open is the community-controlled development platform. In this platform, the community, with the maintainer at the front, controls the agenda and the output of the project. A company may try to influence the direction of the development by sponsoring or providing developers to the project, but the final decision rests within the community.

If a company is unhappy with the progress or future path of the project, it may try to take over the project or create a fork. Nonetheless, in an active project, the company risks being blacklisted or even frozen out, something that will depreciate or negate goodwill gained from OSS involvement.

However, an active community will provide development resources to boost internal developers and bootstrap contributions from the company. The resulting software may increase the functionality and capabilities of the firm’s offerings. This can result in better products for less cost. This is the idealistic concept of OSS, and what many companies envisage when entering the community. Nevertheless, the number of OSS projects that provide these kinds of resources are still few, but for those that are active, companies are able to gain much from them.

### 10.2.3. Hybrid operational platform

Companies with a complex offering may employ several different development platforms, some company-driven projects and some community-controlled. If the company is able to fully utilize the capabilities of the community-controlled platforms, the company-driven platforms may provide a competitive edge.

However, there is a risk of creating friction between different development teams if the methodology in development differs between the projects. It would be interesting to discern if a company-driven OSS development platform is more effective than a normal internal project. Nonetheless, this is beyond the scope of this thesis.

### 10.3. Offering analysis

The discerning offering strategies employed by OSS companies concern the revenue models. There are three distinguishing constructs describing different revenue models, subscription, proprietary licensing, and dual licensing, the latter being a hybrid revenue model combining the first two constructs.

#### 10.3.1. Subscription licensing

The revenue models are directly connected to the offering. Since the definition of OSS is that software is licensed by an OSI-approved licensor and since the most common license, GPL, does not allow for proprietary licensing, most companies offering GPL software offer bundles
of software and services. For this kind of offering, the subscription revenue model is well-suited.

The customer pays for access to the software product, for updating services, and for support. A consultancy firm could offer the same services the company offers, but the subscription model allows for a stable relationship and income, instead of the pursuit of hourly consultancy fees.

Moreover, subscriptions are generally accepted by the OSS community as a legitimate revenue model, and are less likely to alienate the project members.

10.3.2. Proprietary licensing

If the community accepts subscriptions, proprietary offerings based on OSS are frowned upon. A proprietary license is the traditional revenue model for software companies, and the license generally limits the user's rights to the software, something OSS is opposed to. Moreover, the most commonly used OSS licenses do not allow for proprietary licensing. However, the viral properties of the GPL license do not apply in certain cases, depending on how the proprietary software communicates with the GPL-licensed software.

Some licenses, such as LGPL and Apache, allow for proprietary product offerings. The offering of a proprietary revenue model is often in the form of a perpetual user license, where updates can be included or offered as an additional agreement.

Proprietary licensing may also allow the supplier to bundle third-party proprietary products, something that is not legally possible for many OSS licenses. In this way, the final offering can be made competitive and attractive for the customers.

However, the OSS community generally scorns proprietary OSS products, unless the project consents to its usage. More often, companies offering proprietary products based on OSS are accused of exploiting OSS, which can lead to blacklisting and negative publicity. Alienating the community will decrease the benefits that a company may gain from good community relations.

10.3.3. Dual licensing

There are a multitude of suggested revenue models that constitute hybrid models. However, in this thesis the hybrid revenue model denotes a mix between the subscription and the proprietary licensing models. This model is also known as dual licensing.

A company that retains the copyrights for the software can utilize dual-licensing. It is then able to release the product as OSS and still offer a proprietary version to other customers. In essence, the company is creating a community by releasing software that may lead to interest among OSS developers, thereby adding development resources to the company.

However, since copyrights for code contributions are based on the individual’s own source code, every developer’s single line of code is protected by copyright. If a company hires a developer, the copyrights for the development are transferred to the company by a legal contract. If an OSS company is to retain the copyright for its software, a similar legal contract is needed, which may decrease community contributions. One way to attract developers is to offer job opportunities for community developers who sign the agreement.

Aside from generating fewer community contributions, dual-licensing may also alienate the community, for instance if the released version is a less functional or earlier version of the proprietary one. In addition, the impact of community contributions is decreased if there is too large a gap between the released and proprietary versions.
10.3.4. Summary

To conclude, there are 27 combinations possible, based on the nine constructs presented above. These combinations make up the business model configurations that could be found among OSS vendors. This will be illustrated in the next section.

10.4. Cross-case analysis

To further analyze the case companies, the configurations found in each case have been mapped based on the construct matrix in Table 14 below.

<table>
<thead>
<tr>
<th>Business position</th>
<th>Operational platform</th>
<th>Offering</th>
</tr>
</thead>
<tbody>
<tr>
<td>Symbolic link</td>
<td>Company-driven</td>
<td>Subscription</td>
</tr>
<tr>
<td>Product affinity</td>
<td>Community-controlled</td>
<td>Proprietary licensing</td>
</tr>
<tr>
<td>Hybrid business strategy</td>
<td>Hybrid operational platform</td>
<td>Dual-licensing</td>
</tr>
</tbody>
</table>

10.4.1. Cendio

Cendio was founded in 1992 as Signum Support; Cygnus Solutions inspired the founders. This makes it the oldest company in the study (since Cygnus was acquired by Red Hat in 2000). The market in Sweden, especially in Linköping, was not ready for the same offering that Cygnus delivered in the United States.

When Cendio started to offer products, the company had a consistent approach: the products always bundled OSS with hardware and/or proprietary components, such as the Ready-Net server for Internet Service Providers (ISP) and later the Fuego Firewall. When Cendio became an ISV in 2002, the product ThinLinc was created in the same pattern. The product bundles OSS and proprietary components. The main parts of the product consist of OSS software, but the management and control of these are made through proprietary components. The product was initially offered with a traditional proprietary license.

10.4.1.1. Business position

Cendio is an old company, with a long legacy of supporting OSS. The business position has always been one of product-affinity. Even if the company has always offered proprietary products, it has been dependent on OSS. Its current product is even more dependent on OSS than its earlier offerings. Without community efforts, the product would not be possible.

The company’s policy for “paying back” to the community has been to feed back source code and documentation to various OSS projects. When the company was offering hardware based products, the usage of OSS was not controversial, and the relationship to the OSS community was good. This relationship used to be a vital part of the market communication, and Cendio built its corporate brand around its commitment to OSS. However, using a proprietary license on a product that mostly consists of OSS is controversial. Even if Cendio has gone to great lengths to make sure the utilization has been within the legal limits, there have been clashes with community members.

10.4.1.2. Operational platform

One part of Cendio’s business idea is to gain non-paid development resources to save costs for producing its software products. Over 80% of the total code in its products consists of
OSS, while the rest is made up of in-house proprietary components. Cendio has painstakingly investigated how the different components communicate with each other and how the different OSS licenses might influence the final product. The reason that Cendio is able to use a proprietary license is derived from how the components communicate and interact with each other.

However, utilizing several OSS projects in this manner means that the company has almost no control over direction of the projects. The operational platform is therefore community-controlled. The company is dependent on the most important OSS projects progressing in a desired way. This is not always the case, so Cendio has often provided its own developers, and with their contributions, tried to steer some of the projects in the desired direction, something which has not always been successful.

In one project, rdesktop, Cendio had the only active developers and the project maintenance was turned over to the company, which could then control the direction of the project. However, since only Cendio’s developers were active, very little contribution came from the rest of the community. Still, Cendio fed back all development from the project as a good community member. Cendio’s commitment to rdesktop has decreased since 2009.

Another example was TightVNC. After having difficulties, the developers at Cendio created the fork TigerVNC together with two developers, one from Red Hat and the other from the Cendio-sponsored project VirtualGL. These two activities enabled Cendio to gain better control of its most important OSS project (TigerVNC) and an increasingly important project (VirtualGL).

In these two projects, Cendio managed to gain control of the development, but since most of the development was made in-house, the company did not gain the resources it coveted from the community. In addition to these projects, Cendio’s developers still have to develop the proprietary components. This puts a strain on the few development resources Cendio has, which in turn strains the firm’s ability to make the product.

The company debated releasing the proprietary components as OSS, but no definitive decisions have been made.

### 10.4.1.3. Offering

Cendio offered ThinLinc as a proprietary product, with a license and additional upgrading agreements. However, the main customers, public schools, asked to “rent” the software with the upgrades included for a number of years. This became the standard revenue model for the company.

However, Cendio decided to implement a subscription model instead, moving away from the proprietary licenses. Cendio also offered support services along with the software. This was partly due to the concentration of customers in Sweden, and partly due to support issues. The partner strategy was first implemented with the concept of partners handling first and second-line support, and Cendio handling the most severe cases. To this end, Cendio’s partner program had requirements for a minimum number of certified technicians.

However, most partners offered too many products and Cendio was not prioritized, which led to the certified technicians not working with the product and finally not being able to deliver advanced support. Cendio was forced to accept responsibility for first and second-line support in addition to the former third-line. Today, Cendio provides approximately 75% of all support, of which 25% is first-line support.

Thus, Cendio makes use of a hybrid revenue model consisting of a subscription revenue model paired with a proprietary license. This may seem like an odd model, but it is the result of two processes: the customers’ demand to “lease” or “rent” the software, and the process of making the final product OSS.
10.4.1.4. Cendio’s business model configuration

Cendio’s business model configuration is called community utilization. However, there are several misfits between all of the business model factors, and the company is aware of the misfits in its own business model. Nevertheless, the company has been true to its legacy and created the software product in the same way as its earlier hardware offerings. However, where the model worked for bundled hardware offerings, it is not true for software offerings. The reactions from the community surprised management, since Cendio has been accused of exploiting OSS projects. Using an all-proprietary product will always be controversial when GPL licenses are involved.

Cendio’s business model is not sustainable. The company is not able to enjoy all the benefits from community support, and is forced to make use of its own resources when it comes to development. In addition, the company is not large or widely known enough to benefit from any brand benefits from the projects it is involved. A community immersion model, such as the one utilized by CodeWeavers, would probably be a better model, but since Cendio’s offering consists of more than one program, a slightly different model is needed.

Cendios’ management is well aware of these misfits, and some steps have been taken to change the business model. Since the company is dependent on OSS programs, the major changes lie in the operational platform and the offering. The firm has already started a new OSS project together with external developers, so it knows how to set up and manage a project. Cendio’s developers are well-known as competent and capable developers, so when (or if) the company decides to release its proprietary software, the reputation of the developers may benefit the project. In addition, Cendio has changed its revenue model from a proprietary license model to a subscription model. This means no changes for the current customers, and therefore no risk to revenues. Consequently, no changes are needed for the revenue model if the company chooses to become all-OSS.

Will the change help the company to become profitable and attract new customers and community developers? Only time will tell.

10.4.2. CodeWeavers

CodeWeavers, as a consultancy company, became involved in the Wine project in 1999, but changed its business model to become a software vendor in 2002. The company offers an array of proprietary products under the name CrossOver. The software contains proprietary tools to provide a better user experience not included in the community version. The change from a consultancy firm to a software vendor saved the company from failure after Corel withdrew its contract. Over the years, the firm has created a strong bond with the Wine project by hiring the maintainer of the project and making him CTO, as well as by hiring the most active developers of the project. In essence, most of the active developers in the project are hired by CodeWeavers. In addition, CodeWeavers actively sponsors annual worldwide events for the project members with free travel, hotels etc. Moreover, the company actively supports the project by feeding all code back to the project, while the marketing community releases and supports the project and OSS in various ways on the company website.

The management team consists of two programmers, the CEO and CTO, and two non-programmers, the COO and the Vice President of Sales. CodeWeavers is a small company, but it is profitable. The focus of the company is to stay afloat in order to support the staff and the Wine project. The founder and CEO is a devout follower of the principles and philosophies of OSS, which has cultural implications. The longtime goal of the CEO was to return to the original business idea of a consultancy firm which helps other companies to port their Windows software to Linux and Mac, rather than producing a proprietary software product. This has affected how the company has approached the community project.
10.4.2.1. Business position

CodeWeavers’ business is dependent on the Wine project and community contributions for its offering. This product affinity strategy and dependency is evident given the firm’s marketing focus on the community. The preferred communication channel is press releases on the company website. The company reinforces its bond to the community through humoristic press releases. In addition, community news, such as new Wine releases, is also communicated through the company website. Moreover, CodeWeavers fully sponsors the project’s website and the annual developer conferences.

The strategy of targeting marketing towards the community means that Linux customers are also reached. Even if Linux customers represent a dwindling percentage of the firm’s revenue, they are still important for the community. All Linux users, paying or non-paying, are potentially future community developers and a target for recruitment.

Another indication of the close bond to the community is that the company has downplayed its corporate and product brands, and promotes the Wine project brand extensively. However, the company is careful not to overstate its importance for the community, since this could undermine the relationship.

10.4.2.2. Operational platform

CodeWeavers’ products are directly dependent on the company’s progression in the community, and even if the company, including the maintainer, hires a large proportion of the active developers in the Wine projects, the community is still in charge of the roadmap. Wine existed before the company was involved in the project, so the community, and its developers, owns the copyright for the code, and use the LGPL license.

The Wine project community could be considered an integral part of CodeWeavers’ human capabilities, and many of the firm’s developers have been recruited directly from the community. However, this does not mean that the company can direct most of the development in the OSS project, only influence it. In a sense, the company has immersed itself in the community. The relationship between the Wine project and CodeWeavers is reciprocal. While the company is dependent on the community for its products, the community is sponsored and gains marketing from the company. In addition, most of the executives are, or have been, directly involved in the Wine project as developers, clearly indicating that immersion into the community is natural for the company.

One problem with the approach of hiring community developers is that they can be spread all over the world. The Wine project members were predominately European, but some came from other parts of the world. Due to regulations for immigration and other legal issues, aside from the personal issues of leaving one’s native country, CodeWeavers now has employees working at home in different time zones. This causes some communication problems, but has also been an advantage for large releases or defeating significantly stubborn bugs.

The Wine project community is also connected to the wider Linux community, since the purpose of the project is to allow Windows software to run on Linux and Mac. However, Mac users lack the traditional ties to the OSS community as Linux users. CodeWeavers’ Mac products can function as a link between the OSS community and Mac users, which may benefit all parties in the long term.

There is a good fit between the product affinity strategy and the close bond with the community. Even if the company is not able to fully control the advancement of the community project, it can still heavily influence it. However, since the products are so dependent on the community progress, all contributions will affect the quality of the products. Moreover, if the company were to try to control the community openly, the relationship with the non-employed project members might deteriorate, which would negatively affect the products.
10.4.2.3. Offering

CodeWeavers offers proprietary products for Mac and Linux users. In addition, the company offers consultancy services based on its products or the Wine project. The products have been historically aimed towards business customers on Linux, who needed Windows software and tools to run on Linux desktops. However, the Linux desktop market has been stagnant for some time, something which has had negative effects for CodeWeavers. In addition, the individual Linux user was never a primary customer, since such users were generally computer savvy enough to use the free Wine version. The added benefits of CodeWeavers’ products were not attractive enough for the Linux user and not designed with these customers in mind.

The business focus changed when the company made the Mac product. In addition to the Mac version, the company also released a version geared for games. This indicates that the new customers are individuals, not only business customers. Even if the offering still concerns office applications, the company also offers products that can attract individuals. However, the Mac customers are not closely connected to the customer or the Wine project, which creates challenges for the company when it comes to marketing.

The proprietary offering would hamper the relationship between the company and the community if the strategy and operational platform were different. For CodeWeavers, there is a good fit between the offering, the business strategy and the operational platform. However, a subscription revenue model could also work without negatively affecting revenues. Moreover, the proprietary revenue model may be better suited for Mac customers, which are the main clients of the company today.

10.4.2.4. CodeWeavers’ business model configuration

CodeWeavers’ business model configuration could be summarized as community immersion. The model is functional, but it puts limitations on the company in regards to size, profitability and cultural implications that need to be in place, such as a firm belief in OSS. The business model is also dependent on its positive reception by the community. If the community mainly consists of paid developers, or if the community, and specifically the project managers, is suspicious about the firm’s agenda, it will be hard to immerse itself into the community. It is important to understand that the company not only forms bonds with individuals within the community, but also with the community itself (as an entity).

In addition, if the company has more than one project that it wishes to support and immerse itself into, it may put a strain between the different projects and the support from the community members.

The community immersion business model works for a small, ideological firm that is happy to be profitable enough to support the company and the (one) community project. If the company becomes too big or profitable, the bonds with the community may become strained, even reaching a breaking point or perhaps becoming a liability. There is a reciprocal dependency between the product affinity strategy and the close bonds in the operational platform. If the company has enough resources to decrease the dependency of the community, or tries to control the project openly, the business model will fail.

The community immersion model may be one way for an OSS vendor to start and perhaps aim towards the position and business model of Red Hat’s. One company that was on the verge of creating a similar position was MySQL.

10.4.3. MySQL

The one important difference between MySQL and the other cases is that MySQL created its own community by releasing its source code as OSS in 1999. The development in the project
was initially in-house, but the community became interested after the release, and even more so in 2002 when the license was changed to GPL.

Despite the interest, the company was still dependent on the income from its proprietary license, mainly from Windows users. To protect that income, the company needed to retain the copyright within the company. If a community developer was interested in contributing, he or she had to sign an agreement wherein the copyright was signed over to the company. The agreement was controversial and likely meant that fewer developers came to work with the core product than without it.

Nevertheless, MySQL soon became the most popular OSS database and was often used with Linux, Apache and programming languages like PHP and Python. The combination was known as the “LAMP” stack. The timing of the company made its product part of the dot-com boom, where the LAMP stack was widely used among both startups and older companies that created a website. The result is that the MySQL community is much larger than the Wine community, even if the majority of members are not developing the core product.

The co-founders, who were also developers, found investment capital and recruited a management team with the purpose of expanding the company. The interest in MySQL was so great that major American venture capitalists invested in the company. Such investments also spurred more interest in the company and its products, which in turn led to increased product downloads. When the Enterprise product was created, inspired by Red Hat, the challenge was to convert the community users into paying customers. In 2005, the company calculated that just 1 out of 1000 users was a paying customer.

10.4.3.1. Business position

The large number of downloads – over 100 million in 2008 – created a well-known brand, well-versed with the goal to become “first-in-mind” for OSS databases. This “disruptive” business strategy, which was to commoditize the database market and be the market leader on the depreciated market, has startling similarities to many of the unsuccessful business model of the dot-com era. For example, even if the company was slowly converting non-paying users into paying customers, market share was the priority. Market share was believed to directly influence the planned Initial Public Offering to become as successful as Red Hat’s was in 1999. However, the venture capitalists in the company were looking for a successful exit, and the opportunity in 2008 when Sun Microsystems Inc. wanted to acquire MySQL AB was too good to ignore.

The company’s marketing was aimed at four targets: primarily customers of the Enterprise product, and to a lesser extent investors, partners and the community. The company was not interested in converting “non-believers” to OSS, but rather in making sure those who liked OSS would consider MySQL as their first choice for databases, and in converting corporate users into paying customers. After Sun took over, more efforts were made to convert users to customers by differentiating the offerings. However, very little marketing was aimed at the OEM customers, even if these customers were an important source of income early in the company’s history, but their role decreased in the later years.

The large community, and the large number of known installations (over 10 million) built the MySQL brand. In fact, the main utilization of the community for MySQL was for brand recognition. By adding users to the community and knowledge about the brand, the awareness of the company and its product increased, which in turn added (albeit slowly) more paying customers as well as partners. The partner “ecosystem” of MySQL also added professionalism and third party offerings, which further widened the possible applications of the products and thus further grew recognition and awareness of the brand.

The community strategy for the business position was that the community could provide a) brand recognition, b) quality assurance, and c) a recruitment base, in that order. The business
position for MySQL was really a hybrid strategy. Similar to Red Hat, the company primarily utilized the community for brand recognition. However, even if the company was not dependent on community contributions for the product, it was dependent on the quality assurance the community provided. If the company had not been acquired it is possible that the strategy would have turned into a clear symbol link as the company continued to grow and its dependencies on the community decreased.

10.4.3.2. Operational platform

Even if the company was dependent on the community for quality assurance, the company also recruited most of its developers from the community. As a recruitment base for MySQL, the community provided many opportunities to find talented programmers. The company managed to recruit a number of very talented developers this way, so talented that one of the co-founders felt that he would not have made it into the company based on his programming skills had he faced the same recruitment standards. In addition, this meant that the company was in the same situation as CodeWeavers, with the same benefits and disadvantages relative to different time zones.

Since the company controlled the development and releases of the core product and the community was dependent on the releases, the operational platform was company-driven. In addition, community developers interested in contributing to the core product were required to sign agreements to off-hand their copyrights to MySQL. Just like Red Hat, MySQL is in firm control of the development for the product and in the community.

The company-driven platform is also evident in that the executives were separated from the development side, even if high executives like one of co-founders were immersed in development. However, since the community was so important for the strategy of the company, most of the higher executives were accessible for developers. In addition, one vice presidency was set aside for community contacts.

10.4.3.3. Offering

MySQL offered three versions of its product with two different licenses. All offerings were a bundle of different products, which the customer could order directly from MySQL. Partners supplied other configurations and bundles. For customers using the product within their own proprietary product, the company offered a proprietary license and an OEM license for all components. The company offered two subscriptions for customers paying for the OSS-licensed products with support services, Standard and Enterprise, which bundled software, support, upgrade services, and optional training. In 2009, Sun created a “community version” for its community users. This version lacked some of the features of the Standard edition, and was an effort to convert more users into paying customers.

MySQL offered both subscriptions for its OSS-licensed software, and a proprietary license for its OEM software. These revenue models are collectively called dual-licensing, which is a hybrid model. Even though the proportion of income from the proprietary licenses was decreasing and thus controversial, the company decided to keep the option.

10.4.3.4. MySQL’s business model configuration

MySQL’s business model was spawned in the dot-com era, and despite the large community and the takeover by Sun (and Oracle), there are misfits between the business position and the revenue models. As a free entity, the company was never profitable after 2001, and the policy of retaining the copyrights for the product clearly hampered its relationship and usefulness with the community. The question is how long the company would have been able to retain its leadership if it had not been purchased.
When Sun Microsystems acquired MySQL, it was to learn how to generate business from OSS. However, it seems like Sun did not understand or disregarded lessons learned, because when Sun started to change the offering, the founders, the CEO, and several key employees left. What Sun did was to introduce a community version, with less functionality than its commercial version. This was very controversial, and by doing this, Sun started to consummate the MySQL brand in order to gain revenues. The reasoning behind the change may have been the financial crisis, which turned Sun’s business into disarray. While the company was hemorrhaging, Sun made a desperate move to try to “cash in” on the community. However, the changes were not enough, and Sun was for sale. After IBM’s offer was turned down, Oracle purchased Sun in 2009. The question of how Oracle, the new owner, will handle MySQL still remains.

The business model configuration is a different version of the community utilization as employed by Red Hat. However, MySQL’s strategy has several misfits in the business model. In addition, the “disruptive” strategy of depreciating the database market, which has roots in the Dot-com era, is not sustainable over time. At some point, the company had to become profitable, which so many businesses failed to do at the end of that era. For the company to be successful in its endeavor to depreciate that market, it should have discontinued its proprietary offering, and allowed the community to contribute to the core product without signing their copyrights away. Even if no numbers are known, and if the OEM licenses were a rapidly decreasing part of the income and thus less important, the company should have cancelled them earlier. At the time, the company was converting users into customers, and even if the conversions were not fast enough for Sun, the trend was still positive.

The changes Sun made to the business model were so profound, that instead of terming the model community utilization, community exploitation is closer to the mark. The community’s regard for MySQL plummeted, but the question remained. Will Oracle try to regain the community’s confidence for MySQL? Only time will tell.

10.4.4. Red Hat

Red Hat became a public company in 1999, and has offered Linux products with GPL since 1994. The company quickly became the market leader for retail Linux distributions. In March 2003, the company abandoned its original concept of selling retail boxes with Linux and began to focus on Enterprise customers. As a result, all contact with the thriving community and its customers was lost. These groups complained to Red Hat and accused the company to be less accessible than known proprietary vendors. Red Hat realized its mistake and announced that it would create an open project; in November 2003, Fedora Core 1 was released.

Red Hat acquired a number of proprietary companies, but the most significant purchase was Cygnus in 2000. The “new” Red Hat was then the undisputed market leader for OSS. Another important acquisition was JBoss in 2006, which opened up a new market segment, Middleware, for the company as well as a new community. This was unprecedented, since the company had focused on its Linux products. Since then, Red Hat has acquired other companies to enhance its products, and is well-positioned in the emerging “cloud” market with the acquisition of Makara in 2010.

Red Hat is the largest contributor to OSS in the world. Most of the development is done in-house, but the results are fed back to over 100 OSS projects. The most prominent projects are Fedora and JBoss, but Red Hat is also a large contributor to the Linux Kernel project, among many others. However, the majority of projects are connected to the Linux product.

10.4.4.1. Business position

Red Hat’s community interaction is a symbolic link, where the community is utilized as a marketing tool. Red Hat is a large company with many developers, and is the world’s largest
contributor to OSS projects. Even if much of what is produced in the projects will end up in
the commercial product, the lead-time is long, since the life cycle of the products is seven to
ten years. The effects of the symbolic links to the community are not only brand recognition
and awareness, but Corporate Social Responsibility (CSR) as well.

By working with communities through sponsoring and promotion, the company gains
goodwill through CSR. Therefore, the symbolic link with the OSS community is creating
brand recognition and awareness, goodwill, and a communication channel to users. Even if
some beneficiary effects from the community interaction affect the operational platforms and
the offering, the primary effects for the company lie in its business position.

Marketing efforts are mainly concerted towards existing and new customers. Even if Red
Hat has many customers, the majority are not concerned about OSS, something illustrated by
its marketing messages. Opposite to MySQL, Red Hat is trying to convert non-believers to
become interested in OSS. Therefore, the message the company is sending concerns the inno-
vative powers of collaboration, a message very similar to what IBM sent in 2001. As the OSS
market leader, and in order to continue to grow, Red Hat has to widen its market and convert
customers who are skeptical to OSS products. In addition, Red Hat has a stable financial situ-
ation and is therefore reluctant to hurt its brand. Therefore, the company is still in the process
of building its brand and widening market share, preferably by competing with proprietary
competitors.

The company is also addressing partners with marketing; Red Hat has created a partner
ecosystem similar to MySQL’s, if not larger in scale and scope. Moreover, since Red Hat is a
public company, a lot of information (and marketing) is directed towards investors. The mes-
gage of collaboration works well for partners and investors alike.

10.4.4.2. Operational platform

Red Hat provides many OSS projects with development resources. In fact, the majority of
employees develop OSS as their primary business, often in projects not managed by the company. This means it is Red Hat that runs the agenda for the development of its offerings. Even if the results from the community projects are used in its products, Red Hat is in full control of its product.

In addition, the company has distanced itself from its main community projects, since it is important not to be seen as too dominant. When changing its offering in 2003, the company deemed correctly that it had all of the development resources it needed in-house. However, the marketing aspects of the community (and customers) were disregarded. Its community projects were created to provide a channel for the community and customers to communicate with the company, but the contributions were not necessary for its commercial products.

Moreover, the executives of Red Hat were disassociated with the development side. They had been recruited for their ability to run a large corporation, rather than their connections in the community. This reinforces the concept that the company is independent from the community in terms of operational platforms, and that there is a good fit to the business position strategy.

10.4.4.3. Offering

Red Hat offers two commercial software product categories to its customers: operating sys-
tems and middleware. The products are delivered through subscriptions consisting of software
bundled with upgrade services and support, instead of licenses, making the offering non-
proprietary. In addition, all software, except minor tools, is released as source code according
to the GPL license.

The subscription model is a revenue model that is well-balanced between how the custom-
ers are used to paying for software, and the legalities and philosophy of the OSS community.
Therefore, it is no surprise that this revenue model, introduced by Red Hat in 2003, has already been adopted by a large number of OSS software vendors. Use of a subscription revenue model is a good fit for companies that value their relationship with the OSS community, since it is in line with the spirit of most OSS licenses. The company charges for the services it offers, not the use of the software as with proprietary licenses.

10.4.4.4. Red Hat’s business model configuration

There is a good fit in Red Hat’s business model between strategies, operational platform and offerings.

Red Hat’s business model configuration can be summarized as community utilization. The company, as the undisputed OSS market leader, contributes to a large number of OSS projects. Its presence in the OSS market is utilized as a marketing tool, gaining it brand recognition and awareness as well as CSR.

The company is large enough to cater its development needs, and since the products are based on OSS, the company developers are very active, contributing to hundreds of OSS projects. Red Hat is in full control when it comes to ensuring the quality and timing of its commercial product releases.

The subscription revenue model is well-balanced between the needs of the customers and the wants of the OSS community. This keeps both parties content about how the company conducts its business.

The good fit in the business model could explain why Red Hat is profitable and has grown rapidly since 2003. The community utilization with the community as a marketing tool has been successful, but this business model would be hard for competitors to use, something that Novell has experienced. To be continuously successful, it is important that Red Hat retains market leadership, since this is the defining differentiating factor for the company. If the company were to shear its bond with the OSS community, it would be a small player compared to proprietary giants like HP, IBM and Oracle.

For another company to gain the same position as Red Hat, it is possible if the offering is different. For Linux distributors, the advantage of Red Hat is a high barrier to entry, but for other types of software, the same position should be attainable.

However, Red Hat started in a very different position, one where it was more dependent on the community. That business model was different from the current one. One company with a fundamentally different situation, with the corresponding business model, is CodeWeavers.

10.5. Generic configurations

For OSS vendors, the relationship with the community is what sets them apart from other software vendors. The three business model categories, namely the business position, the operational platforms and the offerings of OSS vendors, are all affected by their relationship to the community. The business model configurations of the four studied companies all differ, and it is clear that not all are sustainably profitable. Like all business model concepts, if there are misfits between the business model categories, the effectiveness of the business model is negatively impacted. These configurations are shown in Table 15, below.
In addition, there are two different generic configurations, *community immersion* and *community utilization*. However, only one company uses the community immersion model, while the other three use different versions of the community utilization model. These labels have been used based on the configuration found in the study, and should not be considered tied to them. However, there are situations where a community immersion configuration is in fact community utilization, as addressed in the Discussion section, below.

### 10.5.1. Community immersion

The community immersion model is valid and sustainable, but it has limitations as the CodeWeavers case clearly shows. Aside from support from the individuals in the community, this model is probably only possible if there is only one main project that is immersed. Even if other projects are involved, most of the resources from the company should be aimed at one project.

In addition, the size of the firm is limited as well as the scope of its offerings. Since this strategy is perhaps more suitable for a small company, the number of commercial products would be limited, which also restraints the diversity of customers. An overly broad or faulty-positioned product portfolio could be troublesome for a firm. The products must be well-suited to be attractive for the best customer. If the products offered by the company are attractive for paying customers and the costs of the firm are controlled, a community-immersed company may be viable for a long time. CodeWeavers struggled with a corporate Linux desktop market that never took off, together with reluctant Linux users as consumers; when the company reoriented towards Mac users, it became more profitable.

The community immersion model is not always easy to use; for example, the company can be hampered by community decisions regarding the evolution of the project. For a company dependent on the community, CodeWeavers actively recruits from the community to be able to influence decisions. The benefits for CodeWeavers are several: full access to talented developers, influence on community efforts, goodwill, and other marketing benefits.

The key issue for CodeWeavers is to be attractive for paying customers. In that sense, CodeWeavers may need to focus more on its marketing communication towards customers instead of the community. The marketing efforts made sense when most of the customers were Linux users, but since Mac OS customers are becoming more important for the company, its marketing should adapt. Mac OS users are not connected to the Wine project, and generally not interested in OSS. Could the company’s close immersion with the community be at risk if the communication changes? Could this be a reason why CodeWeavers is hesitant to direct marketing towards its customers?

### 10.5.2. Community utilization

For MySQL, the community relationship meant three things: *brand recognition*, quality assurance, and recruitment base. The company encircled the source code with agreements that repelled rather than invited community contributions. Still, the product was increasingly pop-
ular, and the company started to convert mere users into paying customers. In addition, the company fashioned a vital ecosystem from partners that further attracted customers.

Nonetheless, the strategy to disrupt or break up the proprietary database market and commoditize it came at a price. There is a limit as to how long a company can disregard being profitable before the investors start to complain. The plan for the company was to go public, but perhaps this took too long, so the investors found another solution for an exit. When Sun Microsystems acquired MySQL, the former hardware giant was already struggling. The company had several OSS products, but was unable to produce them at a profit. The acquisition of MySQL was clearly strategic, a way to learn how to conduct OSS business. Then the financial crisis hit, and Sun was stumbling. Perhaps in desperation, the company tried to generate income to stay afloat, and fundamentally changed its business model, moving closer to community exploitation.

MySQL’s business model struggled with the discrepancy between its business position and the offering, where the dual licenses and the need to retain the copyrights repelled community contributions. For the company to fully realize its disruptive strategy, the proprietary business should have been cancelled. This would have sent a more forceful message of the company’s intent, and possibly attracted more investors and customers. Instead, the company ended up being (eventually) sold to the market leader of the market it sought to disrupt.

Another struggling company, Cendio, had traditionally used a community utilization model, which always comprised the bundling of OSS products with hardware and proprietary components. Therefore, it was this concept that was used when the company offered a software-only product. However, this kind of bundling is controversial, since GPL and similar licenses in general make proprietary licensing legally impossible. Nevertheless, Cendio possessed the knowledge to circumvent these problems. The communities are clearly seen as a pool of resources which the company utilizes in its products. The reason for Cendio to use OSS is to lower its cost of development and thus be able to get a larger margin for its products. In short, Cendio is dependent on the community to be able to deliver products.

However, since Cendio was not able to fully control development in the key OSS projects, the company created a fork. Even if the fork itself seems accepted, contributions from outside the company have been limited. In addition, since the important component in the product is proprietary, there are no outside resources to tap for development. Cendio has been able to make use of several OSS programs, but the concept is very close to that of community exploitation, and the company has been questioned about this several times. Releasing the proprietary component or creating a new project may bootstrap the in-house developers, and the company may gain the additional resources it has always coveted. This move is not as controversial as it might seem, since Cendio’s main customers are municipalities and engineering companies that are not interested in operating unsupported software in their live production. In fact, many would probably endorse this move, since this would lessen the risk if Cendio were to close. If this should happen, a third party supplier could be contracted to maintain the software and provide support.

The glaring misfits in Cendio’s business models are something the company is aware of. The firm is contemplating releasing its proprietary components as OSS, and it has changed the proprietary revenue model to a subscription model. Even if Cendio is more similar to CodeWeavers than the other companies in the study, the complexities of having more than one OSS in the offering makes a community immersion model unlikely. The company may still benefit from creating a more clear-cut business model from OSS rather than the current one.

The OSS market leader, Red Hat, operates the only fully-functional community utilization model. For Red Hat, this business model is the legacy of the company, but it is also the differentiating aspect of the company. While its competitors are huge multinational proprietary companies, Red Hat uses OSS, and the innovation collaboration that it brings, as its defining
feature. Red Hat is also the world’s largest contributor to OSS and the engine for many OSS projects. In addition, the majority of the developers at Red Hat spend most of their paid time (and for many their spare time as well) developing for OSS projects.

For Red Hat, OSS is part of the company’s identity and the brand. Therefore, Red Hat has taken the baton to inform and convert customers to make use of OSS. A major marketing effort of the company is to promote OSS and inform non-believers of its benefits.

For Red Hat, OSS means brand recognition, legacy and identity. As the undisputed OSS market leader, Red Hat is first in mind when OSS is mentioned. Red Hat also utilizes the community for quality assurance, but perhaps more important, as a testing facility. The company has full control over its product development, while all testing is done in the community projects. That way, the community enjoys the latest development and features, while its customers benefit from certified and tested software. Moreover, through its community efforts, the company enjoys “free” CSR and the goodwill associated with that. Finally, since Red Hat is the undisputed OSS market leader and its brand is so associated with OSS, the company enjoys recruitment benefits. Talented developers interested in OSS are drawn to the company due to its community efforts and contributions, not to mention the prospect of converting a hobby into a profession.

These benefits are too important for the company to endanger, so the marketing effort tries to build brand rather than consume it. Like MySQL, the benefits for the company are centered on the business position, rather than the operational platform, even if the internal resources clearly benefit from the recruitment of talented developers.

The community utilization model of Red Hat is sustainable and profitable, but it is unclear if any other company would be able to make full use of it, due to the size and brand strength of Red Hat. However, other companies should be able to make use of this business model, so long as their products are attractive to customers and they are interested in the OSS community.
11. Conclusions

This section will conclude the findings of the thesis, addressing and responding to the purpose and the research questions posed in the introduction. The section is divided into two different parts, where the first part concerns the business model framework used to analyze the collected data, and the second part contains the findings from the case companies using the business model framework.

11.1. Business model framework

The use of the term “business model” in OSS contexts has usually been coupled with descriptions concerning the firms’ offerings only, with focus on revenue models and what is offered to the customers (Hecker, 1999; Raymond, 2001; Perens, 2005).

In addition, this is how the term is used in business and among journalists (Shankland, 2000; Asay, 2006; 2007; Clarke, 2010a). However, this usage overlooks important strategic issues, such as marketing strategies and community interaction. Moreover, the operational resources, sometimes the key to understanding why some companies are successful, are hardly ever mentioned.

The academic concept of business models is much wider in theory and covers more aspects on how firms conduct their business, where aspects other than just the revenue models and offerings are covered (Chesbrough and Rosenbloom, 2002; Hedman and Kalling, 2003; Osterwalder et al., 2005). Moreover, several researchers make use of the business model as an analytical tool geared towards understanding how companies add value to their customers and other stakeholders (Amit and Zott, 2001; Hedman and Kalling, 2002; Afuah, 2003). Another important aspect of the academic business model concept is that there exist different concept variations. Some business models are more generic and cover one industry or more, while others give the researcher the choice to compose the business model framework based on the theoretical framework he or she is comfortable with (e.g. Osterwalder et al., 2005).

However, some of the suggested generic business model frameworks, despite their solid theoretical foundations which illustrate the width of the business model, can be too complicated to use in a specific context. Nevertheless, such frameworks are important contributions to the research community since they have deepened understanding of the finer details of how firms conduct business.

In this thesis, an existing business model framework was used in the pre-study. However, the findings included how the framework had shortcomings when dealing with the OSS industry, particularly regarding community interaction, which is vital for all companies in that industry. A new framework was constructed based on the experience and renewed literature studies. This new framework was developed to analyze OSS companies, to include the very important community interaction. This framework, shown in Figure 15 below, consists of three elements: business positions, operational platform, and offering.
This framework was specifically constructed with OSS vendors in mind. The theories used in the business position ranged from those found in industry organizational literature (Porter, 1980; 1985), to strategic decisions (Mintzberg and Waters, 1985), to brand strategies (Keller et al., 2008) and lastly to value constellations (Normann and Ramirez, 1994).

The operational platform contains resource-based views and capabilities (Barney, 1991; 2007; Teece et al., 1997; Eisenhardt and Martin, 2000 as well as community interaction (Dahlander and Magnusson, 2005; 2008).

Finally, the offering incorporates theories from Normann (1975; 2001). The concept of business model configurations, which constitutes generic factors for OSS vendors, draws inspiration from Miller and Friesen (1984).

In addition, despite its specific use in this thesis, the framework was constructed so it may be adapted and adjusted by another scholar to be used in other industries. The author claims that the framework is sufficiently generic in its basic form to work in most industries, since the theories in each element could be changed to fit other purposes.

### 11.1.1. Business positions

In this framework, the business position factors encompass the strategies of the companies, such as marketing, competitor, channel, recruitment, and OSS strategies. These strategies are linked to external factors, for example legislation and industry business logic, and various stakeholders, such as communities, customers, competitors and partners.

The community is vital for these companies, and is unique to the industry, since the interaction may differ between different firms, depending on their strategies. The external factors and the strategies concerning them influence both the operational platform and the offering of the company.

### 11.1.2. Operational platform

The operational platform concerns the resources the companies use to operate, such as organizational, human, financial and physical capital. Typically, the operational platform concerns matters of handling these resources in different ways, which in turn is dependent on the firm’s strategies.
The community is also important for the operational platform, particularly in terms of recruitment, but also for its development and quality assurance resources.

11.1.3. Offering

The offering consists of what the company offers to the market, which in turn is made up of the products offered and how the revenues are collected. The companies studied in this thesis were OSS vendors, with the common denominator that they offered software products based on OSS. Another similarity was that the revenue model used was subscription-based. However, in other aspects, the studied firms differed in terms of licenses and secondary offerings.

The offering of an OSS vendor can also have an impact on the community, since it communicates the company’s intent in a direct way. There have been incidents where changes in an offering have sparked controversy in the community, which forced the company to communicate better, or change its offering.

11.2. Business models

The four OSS vendors were studied through their business strategies, operational platforms and offerings. As expected, the most important parts of the business model configurations concerned how the company viewed the community. This affected the business strategies and the operational platforms the most, and also influenced the offering.

11.2.1. Business strategies

With regards to how the company interacted with community projects, there were two strategies found, symbolic link and product affinity, in addition to a hybrid strategy, which combines the two.

The symbolic link strategy means that the company maintains links to the community, primarily for marketing purposes. The company is not dependent on community efforts to deliver its commercial offerings; the community project mainly creates brand awareness and goodwill through corporate social responsibility (CSR). The company provides paid developers that contribute to different community projects. The company may also institute new community projects for quality assurance and recruitment purposes.

The product affinity strategy is the opposite of the symbolic link. This strategy means that the company is dependent on community projects to be able to deliver its commercial offerings. These companies lack the internal resources, and may have to bundle software from several projects to deliver a complete offering. However, there are companies with this strategy that are connected to only one or two projects.

The hybrid strategy is a mix of the two. The company has the internal resources to deliver the commercial product, but is also dependent on community contributions. This strategy may be volatile for the company, as customer demands may alienate the community and thus degrade the company’s capabilities.

11.2.2. Operational platforms

In order to deliver the commercial product, which was based on community projects, three different operational platforms were found: company-driven, community-controlled and a hybrid operational platform.

The company-driven platform concerns companies that have control over the output of the community, either by providing all contributions to the project (more or less controlling it) or recruiting the most active or vocal contributors in a project. A company-driven operational
platform requires sufficient financial means to employ enough developers from the project or it must provide its own.

The community-controlled operational platform was not found in the sample, but rather in a subset of the hybrid operational platform. The community is in control of the output and direction of the project. A company dependent on a community-controlled operational platform can try to influence its evolution by providing development resources, sponsorship or recruiting vocal members. If this is not possible, the company may try to create a development fork and attract developers to that project.

The hybrid operational platform, which concerns firms that utilize several community projects, bundles the output of these resources into the commercial product. Companies with this operational platform have different tactics for different community projects, depending on the importance of the commercial offering.

11.2.3. Offerings

All offerings of the studied OSS vendors consisted of software products, support and services. In addition, all companies had a number of offerings, mainly consultancy or educational services, aside from their main offering. However, what differed between the companies were the revenue models for the main offering. Three revenue models were found: subscription, proprietary licensing and dual-licensing.

The subscription gives the customer access to a software product, complete with installation and configuration scripts, along with updating services and support. The subscription fee is often paid annually. Subscriptions are independent from software licenses, which make it possible to use with all types of OSS licenses.

On the other hand, the proprietary license is the traditional software license and regulates how the customer is allowed to make use of a software product. Proprietary licenses cede OSS licenses and can therefore not be used with all types of OSS licenses. In essence, software with a proprietary license has a unique license for each company and is, per definition, not OSS. However, some licenses and projects concern libraries (and such) and allow for proprietary licenses.

A dual license revenue license deals with software products that have both an OSS license and a proprietary license for the same software. This is possible if the company retains the copyright for the software. The main purpose for a dual license is to allow OEM customers to utilize the software bundled with other proprietary software. This is not legal for many OSS licenses, which is why dual-licensing was developed. In addition, another version of the dual-licensing concerns subscription for revenue but with a proprietary license.

11.2.4. Business model configurations

The four OSS vendors have configured their business models somewhat differently. These configurations are shown in Table 15 above.

Red Hat is the only company with a pure symbolic link strategy, while MySQL had a hybrid strategy. Both of these companies are larger than Cendio and CodeWeavers, and have internal development resources that can develop the commercial offering. The two small firms are more dependent on community contributions to be able to deliver their commercial offerings.

However, Cendio is the only company that does not make use of a company-driven operational platform. This is because Cendio is dependent on several OSS projects for its offering. Cendio controls some of these community projects, while the community controls others. This is a volatile situation for Cendio, which has been forced to fork one project in order to control the direction of the community evolution. This also means that Cendio is forced to provide
more internal resources to move the development forward. However, Cendio was already contributing to the former project and managed to attract other, outside developers, to the new fork.

Concerning offerings, Red Hat is the only company using the subscription revenue model together with an OSS software license for its main offering. Cendio and CodeWeavers have recently changed their earlier proprietary licensing to subscriptions. However, both of these companies are still using proprietary license agreements for their software products. MySQL used an interesting dual-licensing model, where the same software was simultaneously open and proprietary. The same product could be acquired through a subscription together with the GPL license, or through a traditional proprietary license agreement and fee.

11.3. Generic business model configurations

From the four OSS vendors studied, two different generic business model configurations, community immersion and community utilization, were found. However, only one company uses the community immersion model, while the other three use different versions of the community utilization model. These labels have been used based on the configuration found in the study.

11.3.1. Community immersion

The community immersion configuration should be valid for a majority of small OSS vendors. A company utilizing this configuration seeks to immerse itself into a community project. In addition, it is most desirable to transfer the most proficient community developers into the company as resources in the operational platform. In general, the community immersion configuration identifies project members as predominately research and development resources.

When applying this business model, the firm strives to be a good community member, while offering commercial products to paying customers. This configuration does not prevent the company from working with other community projects, and it is not feasible to be immersed in several community projects concurrently unless they are closely related. However, if the company founder or leader is a well-known figure in the community as a whole, a multi-project community immersion model is possible. A character with the personality and stature of Linus Torvalds or Bruce Perens could be credible in such a situation.

A vital problem to address concerns the control of the community project. A community-immersed company must be able to affect the evolution of the project to avoid forking or losing valuable contributions to the commercial offering. Vigorously recruiting the active and vocal members of the community is one strategy that may overcome this issue.

One important issue is to find and attract profitable customers. The product range must conform to the needs and expectations of the target customers. However, in order for a company to be successful at immersing itself in a project, the market communication with customers may suffer. Therefore, it is important that the company realize that some communication must be addressed outside of the community. In general, customers may not be participating in the community, but may become contributors later.

Within certain limitations, the community immersion configuration could be a stable and perhaps sustainable business model. However, it is important that the company manage to balance the interests of the community and the demands of the customers. This particular business model configuration should be utilized in many smaller projects, perhaps where prominent project members identify a business opportunity.

Only one of the studied OSS vendors, CodeWeavers, utilized a community immersion configuration. CodeWeavers’ business model is sustainable and profitable, but on a small scale.
Due to its business model’s limitations, such as the scope of offering and limited diversity of customers, CodeWeavers was only able to find stability when focusing on Mac customers. Regarding control of the project’s evolution, CodeWeavers counters this issue by actively recruiting the most active developers in the project. In addition, the company has focused on communicating to and from the project, in order to inform and energize the members and in this way be able to present the company mission and vision. This can help steer community contributions into a desirable direction.

11.3.2. Community utilization

If it could be said that the community immersion configuration is more suitable for a smaller company, the community utilization configuration is geared towards larger organizations. In this business model, the company could also work with more than one community. However, in most cases there are one or two main OSS projects that are more prioritized than others.

Compared with community immersion, community utilization is more dependent on internal development resources. This is because the company regards the community primarily as a resource for marketing and a source of goodwill. However, some operational benefits remain, as the community contributes to quality assurance as well as a recruitment base for qualified developers. Therefore, companies utilizing this configuration communicate with the community through regulated channels; this emphasis is seen as benevolent. For this purpose, community utilization companies promote important community personae to managerial positions as community contacts. These characters also double as brand ambassadors for the company within the community.

One important aspect of community utilization is that the open products are usable for a wide array of users, spanning from private individuals to large multinational corporations. The commercial offerings are geared towards larger corporations, with an emphasis on long lifecycles and high service content. In addition, it so happens that large customers also are very active in the community project.

Commercial versions, where the revenue model has been in focus, have often been controversial in OSS contexts. Proprietary licensing, or similar models, such as dual-licensing, has had a tendency to alienate the community. Companies using a dual license, i.e. one open license and one proprietary license for the same software product, have had to retain the copyright for the software and therefore enforce a community contribution agreement, which limited community contributions. However, since these agreements often considered specific core components, the field was open for add-ons that could create a larger ecosystem. The subscription revenue model, where the customer is charged annually for service and support, is a revenue model that conforms to the demands of both customers and community.

The community utilization configuration demands more internal resources to be successful. Even if the community is able to contribute, the commercial offering can be significantly different for open products, since the company is focused on catering to larger corporations. The community is important for brand awareness, quality assurance and as a recruitment base. This configuration is a proven business model, which can be both sustainable and profitable.

For MySQL, the community relationship meant three things: firstly, brand recognition; secondly, quality assurance; and thirdly, recruitment base. The company encircled the source code with agreements that repelled rather than invited community contributions. Still, the product was increasingly popular, and the company started to convert mere users into paying customers. In addition, the company fashioned a vital ecosystem from partners that further attracted customers.

Nonetheless, the strategy to disrupt or break up the proprietary database market and commoditize it came at a price. There is a limit as to how long a company can ignore profitability before its investors begin to complain. The company’s plan was to go public, but perhaps this
took too long, so the investors found another solution for an exit. When Sun Microsystems acquired MySQL, the former hardware giant was already struggling. The company had several OSS products, but was unable to produce them at a profit. The acquisition of MySQL was clearly strategic, i.e. a way to learn how to conduct OSS business. Then the financial crisis hit, and Sun was stumbling. Perhaps in desperation, the company tried to generate income to stay afloat, and fundamentally changed its business model, in the process moving closer to community exploitation.

MySQL’s business model struggled with the discrepancy between its business position and offering, where the dual licenses and the need to retain copyrights repelled community contributions. For the company to fully realize its disruptive strategy, the proprietary business should have been discontinued. This would have sent a more forceful message regarding the company’s intent, and possibly attracted more investors and customers. Instead, the company ended up being (eventually) sold to the market leader of the market it sought to disrupt.

Another struggling company, Cendio, had traditionally used a community utilization model, which always comprised the bundling of OSS products with hardware and proprietary components. Therefore, it was this concept that was used when the company offered a software-only product. However, this kind of bundling is controversial, since GPL and similar licenses in general make proprietary licensing legally impossible. Nevertheless, Cendio possessed the knowledge to circumvent these problems. The communities are clearly seen as a pool of resources that the company utilizes in its products. Cendio’s reason for using OSS is to lower its cost of development and thus be able to capture a larger margin for its products. In short, Cendio is dependent on the community to be able to deliver products.

However, since Cendio was not able to fully control development in its key OSS projects, the company created a fork. Even if the fork itself seems accepted, the contributions from outside the company have been limited. In addition, since the important component in the product is proprietary, there are no outside resources to tap for development. Cendio has been able to make use of several OSS programs, but the concept is very close to that of community exploitation, and the company has been questioned by project members about this several times. Releasing the proprietary component or creating a new project may bootstrap the in-house developers, and the company may gain the additional resources it has always coveted. This move is not as controversial as it might seem, since Cendio’s main customers are municipalities and engineering companies that are not interested in operating unsupported software in their live production. In fact, many would probably endorse this move, since this would lessen the risk if Cendio were to cease operations. If this were to happen, a third party supplier could be contracted to maintain the software and provide support.

The misfits in Cendio’s business models are something the company is aware of. The firm is contemplating releasing its proprietary components as OSS, and it has changed its proprietary revenue model to a subscription model. Even if Cendio is more similar to CodeWeavers than the other companies in the study, the complexities of having more than one OSS in the offering makes a community immersion model unlikely. The company may still benefit from creating a more clear-cut business model from OSS.

The OSS market leader, Red Hat, operates the only fully functional community utilization model. For Red Hat, this business model is the legacy of the company, but it is also its differentiating aspect. While its competitors are huge multinational proprietary companies, Red Hat is using OSS, and the innovation collaboration it brings, as its defining feature. Red Hat is also the world’s largest contributor to OSS and the engine for many OSS projects. In addition, the majority of the developers in Red Hat spend most of their paid time (and many of their spare hours) developing for OSS projects.
For Red Hat, OSS is a part of the identity of the company and the brand; therefore, it has taken the baton to inform and convert customers to make use of OSS. The marketing effort of the company is to promote OSS and inform non-believers about its benefits.

For Red Hat, OSS means brand recognition, legacy and identity. As the undisputed OSS market leader, Red Hat is first in mind when OSS is mentioned. Red Hat also utilizes the community for quality assurance, but perhaps more important, as a testing facility. The company has full control over its product development, and all testing is done in the community projects. That way, the community enjoys the latest OSS development and features, while the customers’ benefit from certified and tested software. Moreover, through the community effort, the company enjoys “free” CSR and its associated goodwill, which can also attract talented personnel (e.g. Bhattacharya, Sen and Korschun, 2008). Finally, since Red Hat is the undisputed OSS market leader and its brand so heavily associated with OSS, the company enjoys recruitment benefits. Talented developers interested in OSS are drawn to the company due to its community efforts and contributions, not to mention the prospect of converting a hobby into a profession.

These benefits are too important for the company to endanger, so the marketing effort is to build brand, rather than consume it. Like MySQL, the benefits for the company are centered on the business position, rather than the operational platform, even if the internal resources clearly benefit from the recruitment of talented developers.

The community utilization model of Red Hat is sustainable and profitable, but it is unclear if any other company would be able to make full use of it, due to the Red Hat’s size and brand strength. However, other companies should be able to make use of this business model, as long as their products are attractive to customers and they are interested in the OSS community.

After analyzing four OSS vendors, it is clear that the main differences between their business model configurations resided in how their OSS communities were handled. The way the OSS vendor viewed the community and its contribution to the company offering determined its strategy, operational platform and offering. In the end, despite the fact that each OSS vendor used different strategies, managed their operational resources differently, and had different offerings, two generic business model configurations emerged: community immersion and community utilization.

11.4. Profitable and sustainable configurations

Two of the four case companies, CodeWeavers and Red Hat, are profitable. The two companies have little in common besides the fact that they are both profitable. Aside from both being American companies, two of their common denominators are that they are able to fulfill the demands from customers and utilize their community. However, their approaches to this are different, and in certain ways almost opposite. CodeWeavers, for its part, is both immersed with its community and is dependent on it for its offering.

Red Hat, on the other hand, utilizes the community, being self-sufficient in delivering the product. The community is important for the company, but more so from the marketing, quality assurance and recruitment perspectives.

While CodeWeavers offers its product to private customers outside of the community, Red Hat targets corporate customers, who often supply development resources to the community. Both companies first struggled to become profitable after they targeted certain market segments. Red Hat’s original offering, a retail box with Linux, was successful, but the profit from each box was small. CodeWeavers, on the other hand, hoped for many years that the market for desktop Linux would explode.
When Red Hat changed its offering in 2002, the product lifecycle of the offering grew from six months to seven years, and the retail box was replaced with a subscription system. By making these changes, Red Hat changed its offering from consumers to corporate customers, catering to their needs for long product lifecycles and support services.

CodeWeavers went the opposite way. After struggling for many years to build products and services for desktop Linux, the firm changed direction in 2005 towards the booming Mac market. This also meant that the focus shifted from corporate customers towards consumers. Since the product primarily supported office applications, the new product was geared towards entertainment, such as games and related software.

In the long run, Red Hat’s business model is more likely to be sustainable. The company is the undisputed market leader and outperforms all of its competitors. CodeWeavers, on the other hand, is a small firm which competes as an underdog compared to its main competitors. However, if the Mac market continues to rise as fast as it has, more companies might be interested in porting their software to that platform. This would also be an option for CodeWeavers. Nevertheless, if that happened, the company would have to change its focus again. As a small and agile firm, however, this should not be an insurmountable obstacle.

11.4.1. Unprofitable configurations

While CodeWeavers and Red Hat enjoyed profitability, the two Swedish companies struggled, but for different reasons.

Cendio employs a community utilization configuration, but this business model is ill-suited for the company. There are several misfits between the different factors, which constitute the problem. First, the company is dependent on the efforts of several community projects for its offering. The firm is too small and lacks internal development resources to fully create the product. Second, the company uses a proprietary license, to which some communities have reacted negatively. However, Cendio has embraced the subscription revenue model, which should soothe some critics. Third, the company is too small and obscure to gain benefits that are connected to the community utilization configuration. The marketing benefits for Cendio are minimal, since most of the projects where the company is able to build brand awareness are too small and not known to customers. In the community projects where Cendio has been active, most activities have come from internal developers, with little or no outside contributions. Some feedback and quality assurance benefits have been detected, but they are small compared with other companies having the same business model configuration. Finally, the financial situation of Cendio has made recruitment impossible.

In order to succeed, the company needs to modify its business model. This can be done in two ways: releasing all software as OSS and making the configuration “purer”, or becoming proprietary. However, the latter would be hard for the company given the current situation, as the product mostly consists of OSS. Moreover, the former strategy would attract some customers, and it should soothe the critics in the OSS community. Cendio has very competent developers, who are well-known in their communities, and becoming “pure OSS” would also benefit them personally. By becoming “pure”, the company will be more attractive to larger organizations that wish to participate with development resources, but are reluctant to do so with the current situation. The move will also make the product less connected to the financial situation of the company, which negatively affects business decisions. This move may also attract and motivate OSS developers to contribute to the product or become job applicants.

MySQL, on the other hand, suffered from a different problem. Despite being one of the most popular OSS companies in the world, with software that was part of the LAMP stack with millions of downloads, the firm was not profitable. In addition, it retained an old business concept from the dot-com era, the “disruptive business model”. The concept was to deliver a full product at a fraction of the price of the established companies in the database mar-
ket, and by so doing “disrupt” the market and gain market share. Therefore, the company utilized the community for three reasons: brand awareness, quality assurance and recruitment. The problem for the company was to convert “free riders” into paying customers. MySQL estimated that one out of a thousand users of the product was a paying customer, and even if the revenues increased, the incomes were still not enough to make the company profitable. To make matters worse, the company made use of a dual license, where the same version was available as a proprietary product. In order to accomplish this, the company had to institute a contributor license, where a community developer interested in contributing to the core product had to sign away the copyright to MySQL. This approach irritated many community members, which limited community contributions and depreciated the brand.

MySQL was acquired twice, first by Sun Microsystems in 2008, and second when Sun was acquired by Oracle in 2010. Sun was hit hard by the financial crisis and tried to change its business model by differentiating between its community and commercial versions. This exasperated the community and the founders of the company, who left Sun and created a fork of the community project. When Oracle took over Sun, it kept the business model from Sun, but angered the OSS community in other ways, which in the end might adversely affect MySQL.

The business model of MySQL stands untested. If the company had remained independent and proceeded with the planned IPO, perhaps it would have become profitable in the end. MySQL had a unique position, and it is not a given that any other OSS vendor might achieve a comparable position again.
12. Discussion and future research

This thesis only encompasses four OSS vendors, carefully selected to represent different aspects of the OSS software market. Other researchers have also studied most of these companies (e.g. Dahlander and Magnusson, 2005; 2008; Campbell-Kelly and Garcia-Swartz, 2010), which indicates the interest of the said cases.

This chapter consists of two sections, where the first discusses the result of this thesis, while the other suggest possible further research.

12.1. Discussion

The most important contribution comes from the generic business model configurations, which were reduced from four specific configurations to two, community immersion and community utilization. Here, the control mechanisms as proposed by Ouchi (1979) are clearly at play. For small companies, community immersion makes sense since it employs the clan control mechanism (Ibid.), where the company employs staff with similar beliefs, competence and experience directly from the OSS community. A close connection with a community project enables the company to enjoy “preferred” status, and thus evaluate potential employees in advance, based on their performance in the projects. With a small, flattened organization with few external stakeholders, it is vital to convey a message to the community about the company’s values, beliefs and that the “ceremonial rituals” (i.e. how work is being done) (Ouchi, 1979) conform to those of the community project.

The larger the company becomes, the more external outsiders influence the company, with demands for further control, which in turn creates a need for bureaucratic control mechanisms. The demands can be regulations from government or external organizations, for example stock exchanges. As Ouchi mentions, most companies have several control mechanisms at play simultaneously (1979, p. 845), but the impact on the organization differs. Red Hat has managed to create watertight compartments between its management and developers, where the OSS communities (and the employed R&D staff) are managed like a research lab in Ouchi’s examples (p. 843). Thus, the strategy of community utilization makes sense, and the company is able to conform to the bureaucratic demands from one set of stakeholders, and the informal, ceremonial rituals of another set. However, for MySQL the transition from being a small company to a larger company was rough, and management decisions, based on rationality but also stakeholder demands, often raised the ire of community members. In addition, the organization of MySQL could not withstand the demands of a larger, more bureaucratic organization like Sun. The founders, and the most influential community personalities, left the company one after the other.

Even if the interaction with various OSS communities differs in the studied cases, it is clear that the OSS vendors invest heavily in R&D along with the OSS communities. This is in line with the findings of other researchers (e.g. Fitzgerald, 2006; Dahlander and Wallin, 2006; Ågerfalk and Fitzgerald, 2008), who point out that there is a shift towards more business-oriented communities with paid developers, which are influenced or controlled by firms. However, this contradicts the traditional concept of OSS and how other researchers portray OSS communities (e.g. von Krogh et al., 2003; Dahlander and Magnusson, 2005). Furthermore, the contracts MySQL made with community contributors further negate this view. However, the community immersion configuration found in CodeWeavers is still true to the traditional view of OSS communities, but it is the only company in this study that employs it. Generally, these changes in how OSS is developed seem to have evolved over time, suggesting a change in the OSS concept; Fitzgerald suggests the term OSS 2.0 (2006).
Moreover, the findings seem to confirm the hypothesis of a convergence with proprietary software, as proposed by Campbell-Kelly and Garcia-Swartz (2010), at least in terms of R&D expenditures (and acquisitions). However, there is a fundamental difference between the R&D spending of Red Hat and Microsoft, in that most R&D efforts by Red Hat are shared with the community, as Red Hat provides developers in hundreds of OSS communities, while the efforts of Microsoft are still retained within the company (see West and Gallagher, 2006). As for the acquisitions, most them have not been incorporated in its commercial products and seem uncoordinated (Asay, 2009b), but all of the source code has been released as OSS, which is also a fundamental difference from most proprietary companies. In addition, Red Hat’s acquisition of Jboss might have been a defensive move against proprietary acquisition, since companies like Oracle had tried to acquire it (e.g. LaMonica, 2006b).

The question is whether there are configurations other than those studied that could also be profitable and sustainable. Only two of the studied companies have been profitable over time, CodeWeavers and Red Hat, and only the latter has been very profitable. However, the reason that Red Hat changed its offering in 2003 was because profits were slim for its original offering, which somewhat contradicts Campbell-Kelly and Garcia-Swartz’s description that the offering was profitable (2010, p. 234). In the same article, the authors seem to have misunderstood Red Hat’s subscription revenue model, referring to it as a “dual-license” (p. 235) and a software license (ibid.). The subscription is within the rights of GPL, which is included with the product. The subscription regulates the services provided by the company, such as packaging, updates, and support. The source code is available, and it is not complicated to remove the brands from the product. Red Hat’s competitors are able to offer the software for free, such as CentOS, or a competitive product such as Oracle Enterprise Linux. Red Hat does not offer services for any other products other than to its paying customers. This is regulated through a service level agreement (SLA), which is not a software license but rather an agreement for services.

Moreover, the dual-license model used by MySQL also seems to have been somewhat misunderstood (Ibid, p. 239). MySQL offered two commercial offerings, one subscription (similar to Red Hat’s) with an OSS license and one OEM (embedded) version with a proprietary license. Identical software having two different licenses, one OSS and one non-OSS, is what constitutes a dual-license (e.g. Välimäki, 2003, Olson, 2006). MySQL was able to offer a proprietary license because the company retained the copyright for its software through contributor licenses. Red Hat does not retain its copyrights for Linux, and is thus unable to use a proprietary license that qualifies as a dual-license.

The two generic profitable and sustainable configurations found in this study are likely to be accompanied by others in a larger sample. One speculative configuration for community immersion that could be feasible is the combination of product-affinity strategy, community-controlled operational platform, and a subscription revenue model. A more “community friendly” revenue model should be easier for a commercial entity to suggest for the community.

Note that there could be a discrepancy in the labeling of the generic business models. The same community immersion configuration as CodeWeavers, i.e. the combination of product-affinity strategy, community-controlled operational platform and a proprietary license, can also be a community utilization strategy. The shift from immersion to utilization relates to the size and ideology of the OSS project. A small firm may very well be able to create products based on the results from a large community, without being immersed in it. However, the number of projects large enough and with a forgiving philosophy – based on how open the project is on proprietary versions of the software – are still relatively few, despite the predictions some scholars (e.g. Lerner and Tirole, 2005; Fitzgerald, 2006). Even so, one successful example is the Apache project, which is a foundation hosting a number of projects based on
the Apache web server software. The license allows for proprietary licensing, and the software is used in a large array of constellations, OSS as well as proprietary (e.g. IBM, 2011). Yahoo, which is a major contributor to Apache, decided to discontinue its own proprietary version and contribute to Apache’s open version (Baldeschwieler, 2011).

The possibility for an OSS vendor to gain the same position as Red Hat is, as already discussed, limited. However, for large corporations with several business models, a similar position is feasible to attain. If a corporation decides to release its proprietary software as OSS, and provide most of the development without restraining community contributions (like MySQL), a bustling community should be the result, not to mention having an attractive offering for both non-paying users and paying customers. Similar scenarios have been discussed by researchers (e.g. Bonaccorsi and Rossi, 2003b; Harhoff et al., 2003, West and Gallagher, 2006). The benefits gained from this community, such as brand awareness and recognition and CSR, are the benefits that Red Hat enjoys today. Given time, the community could also provide product innovation and quality assurance, as well as function as a recruitment base for developers. These possibilities were evident for MySQL, but could have been even more effective had its configuration been slightly different.

Based on the conclusions from this research, the configurations could be applied to current software companies. For example, a popular online service (and company) like Facebook could prosper even more if it released its software as OSS. The massive amount of users and the plenitude of applications are signs of a lively community. Nevertheless, the controversies about new designs, leakage of personal information and other forms of unwanted exploitation could be solved if the software were released as OSS. The large number of users surely contains a significant number of talented developers, who would love to improve the source code. Moreover, “Linus’ Law”, which was proposed by Raymond (2001) and states “given enough eyeballs, all bugs are shallow”, could be very effective against outside threats. In addition, an open culture would allow normal users to test and verify new features before they are released in “the wild”. In fact, an OSS version of Facebook already exists; the Diaspora project was created because of the flaws in Facebook (Wallström, 2010).

12.2. Future research

Every study has its limitations, either in terms of resources or time or both. In this thesis, sometimes tantalizing tracks and patterns of other research have emerged, which had to be put aside in order to finish the study. Other limitations could concern scope or lack of information, something that other researchers may have access to, or are able to collect. This section contains some of the limitations and tantalizing patterns that, according to the author, ought to be studied further.

12.2.1. Scope of the study

The findings in this study are limited in scope, and a larger sample should be able to clarify how common or general these business model configurations are “in the real world”. Another aspect would be to investigate changes in the configurations over time, in order to study how company size, community and influential customers affect business decisions. Moreover, since this thesis has been explorative, perhaps survey questionnaires can be constructed and the results validated through quantitative methods.
12.2.2. Customer and community reactions

This study has been supplier-centric, which leaves out both customer and community perspectives on how business model configurations influence companies in their choice of products or communities to join. Even if studies of developer motivation exist, most are fairly old and OSS has since changed. Customer decisions are even less researched, which further impairs the general understanding of OSS.

12.2.3. Community management and open innovation

Even if this thesis only brushes upon the subject of open innovation, the way OSS companies manage OSS communities should also influence how innovation within OSS projects is created. Is the community utilization configuration that Red Hat employs beneficiary for innovation, or is the community immersion configuration, used by CodeWeavers, more efficient from an innovation perspective? The results from this thesis indicate that the company's business model also affects its community interaction and management, which show the complexity of the relationship between important innovative stakeholders.

12.2.4. Brands

Brands have been mentioned in this study, and it is the author's firm view that this subject warrants further exploration. The effects of branding on B2B customers, the primary customers for most OSS vendors, are fairly well researched, however important questions remain. What happens when you add OSS communities into branding measures? Will that affect traditional branding strategies towards B2B, or will the companies utilize branding strategies conventionally used for B2C?

12.2.5. Business model configurations

The debate about strategy and business models continues. The problems with different denotations of the term “business model” in academia and among professionals are also likely to continue. In this thesis, the use of “business model configurations” is one suggestion to make it easier for researchers and practitioners to discern between “everyday use” and academic use. An alignment of the content in business model frameworks for research is probably not desirable; the risk for similar entrenchments in strategy research would follow. Instead, an alignment in naming the resulting descriptions of analyzed subjects is more feasible. As in this thesis, naming the result of an analytical framework a business model configuration instead of a business model could avoid confusion between researchers and practitioners. Even if the theories used, or factors involved, are of “academic interest” for a practitioner, the result can be useful and appreciated, no matter how it was constructed.

12.2.6. Transitions of business model configurations

Two of the case companies, Cendio and CodeWeavers, were forced by a recession to change their entire business models in 2002, and both became OSS vendors. This indicates a pattern of how business model configurations can change over time. The other two cases, MySQL and Red Hat, made adjustments; these mostly concerning their offerings, but this did not affect the other business model elements in such a way that it would indicate a change in the whole business model. In four cases, two different patterns were found, which could indicate that there are other transition patterns. Further research in finding, identifying and studying
business model transitions would increase the understanding of how firms adapt to and survive in a changing business landscape.
References


Asay, M. (2009d). Red Hat: From manic acquisitions to focused execution. (HTML) Availa-

Asay, M. (2010a). Microsoft's past - the future to Android's iPhone victory. (HTML)


199202274> (2010-12-02).


Vol. 43 (2-3), pp. 143-145.


<http://developer.yahoo.com/blogs/hadoop/posts/2011/01/announcement-yahoo focusing-on-


Bank of Finland (2010). MySQL OEM licenses. (PDF) Available:
<pss.bof.fi/Documents/MySQL_extra_licenses_ORDERING_FORM.pdf> (2010-12-09).

ment, 17(1), 99-120.

cation. Upper Saddle River.


Approach for Creating New Business Models: The General Motors OnStar Project. Interfac-
ces, Vol. 32 (1), pp. 20-34.

Barrett, V. (2009) Why Oracle Won't Kill MySQL. (HTML) Available:
<http://www.forbes.com/2009/04/20/mysql-marten-mickos-technology-enterprise-tech-
mysql.html> (2011-11-17).


Cendio (2010f). *Mix Platforms*. (HTML) Available:


LaMonica, M (2006a). Sun picks GPL license for Java code. (HTML) Available:

LaMonica, M (2006b). Red Hat scoops up JBoss. (HTML) Available:


Lettice, J. (2002). “GPL Pacman will eat your business, warns Gates”. (HTML) Available:
<http://www.theregister.co.uk/2001/06/20/gpl_pacman_will_eat_your/> (2006-02-21).


LinkTech (2010) Välkommen till LinkTech. (HTML) Available:

Linux Foundation (2010). Linux Standard Base (LSB). (HTML) Available:


Red Hat (2011b). *Red Hat Executive Team.* (HTML) Available:


VirtualGL (2010b). Introduction. (HTML) Available:


## Appendix 1: Glossary

This section concerns the abbreviations and terms used in the thesis.

**BM** – *Business Model*, a description or academic model for how companies function.

**Community** – the collective name for user networks often used for all users and developers working with Free or Open-Source Software.

**BSD** – *Berkeley Software Distribution*, a public domain license created at the University of California, Berkeley and used in the original UNIX operating system.

**CLA** – *Contributor License Agreement*, an agreement that defines the ownership of the intellectual property of an OSS.

**Cloud computing** – a term describing software and services that are available through the Internet, rather than on local computers and servers.

**Copyleft** – a play on words for how GPL makes use of the United States copyright laws.

**FS** – short for *Free Software*, a term minted by Richard M. Stallman for the idea of free software.

**F/OSS** – short for *Free and/or Open-Source Software*, a term used when the philosophies (for FS or OSS) in not an important factor, often used by researchers.

**FLOSS** – short for *Free, Libre, Open-Source Software*, a term used by Rishab Aiyer Ghosh to encompass all kinds of Free and Open-Source Software.

**GPL** – *General Public License*, a licensing agreement created by Richard M. Stallman in order to facilitate the idea of free software. The most-used license for free and open-source software.

**GUI** – *Graphical User Interface*, an interface that makes interaction with the computer easier, making use of peripherals like mice instead of only the keyboard.

**ICT** – *Information and Communication Technologies*, the convergence of telephone and computer networks; may also refer to IT.

**IHV** – *Independent Hardware Vendor*, an independent supplier of a hardware product.

**ISV** – *Independent Software Vendor* an independent supplier of a software product.

**IT** – *Information Technology*, a term to describe the use of electronic devices to communicate over other media, such as telephone or radio.

**License** – a legal agreement concerning usage of copyrighted material, such as music, literature, and software.

**LGPL** – *Library (or Lesser) General Public License*, a license agreement for libraries, with less strict rules than GPL.

**Maintainer** - a term used for the copyright holder or project manager (usually promoted by the original copyright holder) of an OSS project. The maintainer is responsible for the development of the project and often controls the direction of the development.

**Software Libre** – a hybrid Spanish name for free software.

**OEM** – *Original Equipment Manufacturer*, a company that has produced a product which is included in another company’s offering and sold under the latter’s brand.

**OSS** – *Open-Source Software*, a term minted by Eric S. Raymond and Bruce Perens denoting a more commercial use of free software.

**OSS Company** – a firm providing goods or services based on OSS.

**OSS Vendor** – a company providing software products based on OSS.

**SaaS** – *Software as a Service*, another name for “cloud computing.”

**TCO** – *Total Cost of Ownership*, a method to calculate cost of software and services for a company.

**VAD** – *Value-Added Distributor*, a term for an Oracle partner with responsibility for a geographical area.
**VAR** – *Value-Added Reseller*, a reseller that offers service on top of selling a product, such as service or support.

**VDI** – *Virtual Desktop Infrastructure*, a computer model for remote desktops on computers.

**VNC** – *Virtual Network Computing*, software for remote desktop sharing.
Appendix 2: List of interviews

Andersson, Stefan, CEO, RedBridge AB, November 1, 2006.

Arnö, Kaj, Vice President Database Community, Sun Microsystems Inc, September 30, 2009. (Phone interview).

Axmark, David, Founder and Vice President, MySQL AB, February 28, 2006 and October 12, 2007.

Igheimer, Johan, CEO, Cendio AB, October 6, 2009 and December 8, 2009.

Larsson, Lars, COO RedPill AB, November 1, 2006.

Mickos, Mårten, CEO, MySQL AB, October 28, 2005

Moosberg, Peo, CEO (former), Cendio AB, March 10, 2006


Rubazkin, Bert, CTO Sun Microsystems Svenska AB, October 6, 2006.

Svensson, Magnus, CEO Red Hat AB, September 30, 2009. (Phone interview).

Spector, Albert, Dr., CTO and Vice President of Strategy and Technology, IBM Software, October, 26, 2005.

Stern, Hal, CTO Software, Sun Microsystems, November 5, 2005 (Phone interview).


Wallin, Inge, CTO and founder, Cendio AB, March 10, 2006

Appendix 3: Interview guide 1 (Swedish)

Bakgrund

- Hur länge har du arbetat på Företaget?
- Hur länge har du arbetat i den rollen som du nu innehar?
- Var arbetade du tidigare?
- Har ditt arbete förändrats under det senaste året?
- Hur ser du Företagets roll i förhållande till era kunder/andra bolag?
- Hur har Företaget ändrats genom åren?
  o Organisation
  o Kultur
  o Produkter
  o Erbjudanden
  o Strategi
  o Kunder

Framtid/Vision

- Vad ser du som Företagets affärsidé/modell?
- Vad ser du som Företagets kärnkompetens?
- Kan du berätta var du ser Företaget i framtiden (i förhållande till t ex kunder, dig själv etc.)?
- Vilken vision har du i förhållande till detta?
- Har ni diskuterat framgångsfaktorer? Vilka?
- Hur ser du på öppen källkod och dess framtid?
- Identifierar du Företaget som ett öppen källkods-företag?
- Öppen källkod jämfört med proprietära program?

Organisation/Struktur

- Skulle du kunna beskriva organisationen utgående från din position?.
- Vem ansvarar för produktutveckling och identifiering?

Roll

- Är du klar med vilken roll och vilket ansvar du har?
- Tycker du att det är svårt att se var ditt ansvar slutar och andras börjar?
- Hur interagerar din roll/du/din uppgift med andra roller? Med andra delar av organisationen?

Teknologi

- Var det så att tekniken möjliggjorde sådant som ni funderat på eller drev tekniken nya lösningar?
- Vilka fördelar respektive nackdelar har öppen källkod?
- Vilka karaktäristika har öppen källkod som gör det lönsamt/intressant?
Erbjudanden

- Kan du beskriva ett erbjudande?
  - Vad är det?
  - Vad innehåller det (t.ex service produkt)?
  - Vem skapar nya erbjudanden?
- Hur säljer ni/marknadsför ni era erbjudanden (eller produkter)?
- Hur resonerade ni runt era erbjudandena?
- Vilka är de element i era erbjudanden som du anser är mest intressanta, värdeskapande etc jämfört med ”vanlig” programvara?
- Hur genomfördes erbjudandeidentifieringen?
- Hur hanterar ni licensieringsproblematisken?
- Hur resonerar ni kring era intäktströmmar?

Process

- Har ni tydliga processer? (Vilka och vad?).
- Hur skulle du definiera en process?
- Hur identifierar ni era kärnprocesser?
- Vilka processer finns du med i?
- Hur ser inlärningsprocessen ut?

Kunder/Leverantörer

- Hur identifierar ni era kunder?
- Känner du att ni är en kundfokuserad organisation?
- Vilka processer har ni med (i samarbete). kunder?
- Vilket förhållande har ni med era leverantörer?
- Hur och vilka hanterar (vårdar). era kund- respektive leverantörsrelationer?
- Beskriv hur kunden när er organisation?
- Hur fängar ni kundens behov?
- Särskiljer ni på stora respektive små kunder och i så fall hur?
  - Segmentering
  - Positionering
- Kan du identifiera var och hur ni skapar värde för era kunder?
- Vilka idéer har ni i framtiden för kunder respektive leverantörer?

Anställda/resurser

- Vilken ”sorts” anställda har ni?
- Hur är mixen gammal/ung bland anställda?
- Hur fördelas resurser?
- Ser du några problem med resursfördelningen?
- Vilka resurser finns respektive finns ej inom företaget?
- Var ligger fokus: produkt eller försäljning?
- Hur säljer ni?

Ledarskap/Kommunikation

- Vilken roll anser du att ledningsgruppen har?
• Hur förankras beslut i organisationen?
• Hur kommuniceras direktiv uppifrån?
• Hur sker kommunikation horisontellt, mellan både individer samt funktioner?

Övrigt

• Support system
• Mätning:
  o Hur mäter ni kundnöjdhet?
  o Hur hanteras återkoppling till berörda?
• Nytt värdeskapande:
  o Hur diskuteras detta?
• Kultur
• Vilka konkurrenter har ni?
• Vad är era konkurrensfördelar?
  o Jämfört traditionella konkurrenter?
  o Jämfört OSS-konkurrenter?
• Hur positionerar ni er på marknaden?
Appendix 4: Interview guide 2 (English)

Internal strategic issues

1. Please describe your business model regarding OSS.
2. How important is OSS for your company today? (In terms of revenues, marketing, research and development or other.)
3. What were the driving forces behind your move to/towards OSS? (Internal – external, primary, secondary, tertiary etc.)
4. How are you working with OSS strategically today? (Organization, management, development etc.)
5. Does your product development of OSS differ from your proprietary software?
6. How do you decide which project(s) are to be OSS and which project(s) are to be proprietary? (For example IP, patents, third parties IP and patents, etc.)
7. Is there any software (owned by your company) you think is unlikely to be revealed (if any)?

External strategic issues

1. What do you think is the greatest threat to OSS (if any)? (Includes software patents, licensing, and marketing campaigns.) Please answer both generally and specific for your company (if possible).
2. Some claim that OSS will be the norm of all software in the future. Is this reasonable, plausible or even desirable?
3. How do customers respond if you offer them solutions based on OSS?
4. What is your opinion of the competitive advantage of OSS (if any)? Please answer generally and specifically regarding your company (if possible).
5. How do you perceive the service of OSS?
6. Is there any market that you see having greater opportunities for OSS than others? Please answer both generally and specifically for your company (if possible).

Industry formation

1. Policies and standards are starting to emerge regarding OSS. Does this affect your company in terms of strategic assessments?
2. The formation of open standards is one part of these policies. Can you describe how your company works with open standards (if possible)?
3. Some countries, like China and South Africa, implement policies to use OSS (mainly) in order to strengthen local software development. How do you think this might affect your company? Please answer both generally and specific for your company (if possible). (Short and long term.)
Appendix 5: Interview guide 3 (English)

Marketing/sales

- What is your business idea/mission?
- What is your outstanding mission? (Overall, per department.)
- What are your goals? (Short term + long term.)
- Do you have a branding strategy? If so, how is it formulated? (Towards paying customers and community.)
- Who are your competitors? Biggest threat (of all). OSS competitors etc.
- Describe your corporate culture. (Does it differ depending on department?)
- Name some of your official customers. (The ones you CAN mention.)
- How do you market/promote your products? (Commercial/non-commercial.)
- Name your most important partners. (If possible, mention in what way they are important.)

Operations/development

- How do you manage (do you?) your in-house development? (Version handling, etc.)
- Does your management differ between commercial and non-commercial development?
- Do you have any significant developer not employed by you working with your projects?
- What are your experiences regarding revenue models and development management?
- Describe your corporate history. (Founding, business etc.)
- Do you own patents and/or copyrights, and if so, how are they defended?
- What kind of agreements do you have? (EULA, partnership, certification etc.)
- How is the company organized? (Departments etc.)

Product/service offering

- How is your customer informed regarding price, offerings, etc?
- Describe your products and services. (What do you offer your customers?)
- How do you charge your customers? (Revenue models, licensing, subscriptions etc.)
- How is your software distributed to your customers? (Downloads, media etc.)
Appendix 6: Phone survey

1. Har företaget en varumärkesstrategi?
   Ja ☐ Nej ☐ Vet ej ☐
   Om ja, beskriv den (kortfattat):

2. Har företaget en marknadsföringsplan?
   Ja ☐ Nej ☐ Vet ej ☐
   Om ja, beskriv den (kortfattat):

3. Var verkar företaget idag (lokalt, regionalt eller globalt)?
   Lokalt ☐ Regionalt ☐ Globalt ☐

4. Har företaget en uttalad strategi kring marknadsexpansion?
   Ja ☐ Nej ☐ Vet ej ☐
   Om ja, beskriv den (kortfattat):

5. Vilken typ av kunder har företaget?
   Organisationer* ☐ Blandat ☐ Konsumenter ☐
   * Företag och organisationer med eget organisationsnummer

6. Finns företagets mjukvaror tillgängliga som öppen programvara?
   Ja ☐ Nej ☐ Vet ej ☐
Om ja, vilka:


7. Äger företaget upphovsrätten för sina mjukvaror?

Ja  Nej  Vet ej

Om ja, vilka:


8. Finns det en plan för att öka sitt åtagande kring öppen programvara?

Ja  Nej  Vet ej

Beskrivning av anledningar (kortfattat):


9. Hur fördelar sig företagets resurser? (%)

Råvara  Hårdvara  Mjukvara  Finanser  Kompetens  Vet ej

* Skog, mark, olja etc.

10. Hur tar företaget betalt (intäktsmodell)? (%)

Licenser  Hyra/lease  Prenumerationer  Serviceavtal  Per timme  Annat

Licenser och Hyra/lease är proprietära mjukvaror som säljs eller hyrs ut. Prenumerationer kan vara en kombination av proprietära eller öppna programvaror som levereras tillsammans med tjänster i olika former. Serviceavtal kan även vara supportavtal. Annat kan vara utbildning, eller fastprisåtaganden, etc.

Om annat, exemplifiera:
11. Planerar företaget att ändra sin intäktsmodell?

Ja  Nej  Vet ej

Om ja, beskriv hur (kortfattat):

12. Hur säljer företaget, själva eller via partners?

Enbart egna resurser  Enbart via partners  Vet ej

13. Planerar företaget att förändra sin försäljning via partners?

Ja  Nej  Vet ej

Var vänlig motivera (kortfattat):

14. Vilken typ av erbjudande(n) har företaget?

Enbart Varor*  Enbart tjänster  Vet ej

* Enbart varor omfattar även paketerade tjänster och mjukvaror.

15. Planerar företaget att förändra tjänsteinnehållet i erbjudandena?

Ja  Nej  Vet ej

Var vänlig motivera (kortfattat):
16. Vilken typ av leverantör är företaget?*

<table>
<thead>
<tr>
<th>Leverantörers leverantör</th>
<th>Leverantör</th>
<th>Kundnära</th>
<th>Annat</th>
</tr>
</thead>
</table>


Om annat, beskriv hur (kortfattat):

17. Finns det en uttalad strategi att förändra företagets position i värdekedjan?

| Ja | Nej | Vet ej |

Om ja, beskriv hur (kortfattat):

18. Vilka är företagets (tre) främsta konkurrenter?

19. I vilket förhållande står företaget gentemot den främste konkurrenten?

| Mkt. svagare | Mkt. starkare | Vet ej |

20. Hur uppfattas företaget av företagets kunder?

| Mkt. positivt | Mkt. negativt | Vet ej |

21. Hur uppfattas företaget av företagets användare (ej betalande, i community)?

| Mkt. positivt | Mkt. negativt | Vet ej |

22. Hur uppfattas företaget av företagets leverantörer?

| Mkt. positivt | Mkt. negativt | Vet ej |
23. Hur uppfattas företaget av företagets konkurrenter?

<table>
<thead>
<tr>
<th>Mkt. positiv</th>
<th>Mkt. negativ</th>
<th>Vet ej</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

24. Päverkar företagets varumärken kunderna?

<table>
<thead>
<tr>
<th>Betydelselöst</th>
<th>Avgörande</th>
<th>Vet ej</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

25. Päverkar företagets varumärken användarna?

<table>
<thead>
<tr>
<th>Betydelselöst</th>
<th>Avgörande</th>
<th>Vet ej</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

26. Hur päverkar företagets varumärken leverantörerna?

<table>
<thead>
<tr>
<th>Betydelselöst</th>
<th>Avgörande</th>
<th>Vet ej</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

27. Päverkar företagets varumärken konkurrenterna?

<table>
<thead>
<tr>
<th>Betydelselöst</th>
<th>Avgörande</th>
<th>Vet ej</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

28. Finns det en strategi för användar- och utvecklarsamfund ("communities")?

<table>
<thead>
<tr>
<th>Ja</th>
<th>Nej</th>
<th>Vet ej</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Om ja, beskriv hur (kortfattat):

<p>| |</p>
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

29. Är communities en del av företagets marknadsföring?

<table>
<thead>
<tr>
<th>Ja</th>
<th>Nej</th>
<th>Vet ej</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Om ja, beskriv hur (kortfattat):

<p>| |</p>
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

30. Päverkar communities företagets varumärken?

<table>
<thead>
<tr>
<th>Betydelselöst</th>
<th>Avgörande</th>
<th>Vet ej</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

31. Päverkar communities företagets marknadsföring?

<table>
<thead>
<tr>
<th>Betydelselöst</th>
<th>Avgörande</th>
<th>Vet ej</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

32. Päverkar communities företagets upphovsrätt?

<table>
<thead>
<tr>
<th>Betydelselöst</th>
<th>Avgörande</th>
<th>Vet ej</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

33. Finns det belöningssystem för användarbidrag?

<table>
<thead>
<tr>
<th>Ja</th>
<th>Nej</th>
<th>Vet ej</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Om ja, beskriv hur (kortfattat):
34. Planerar företaget att förändra kontakten med communities?

<table>
<thead>
<tr>
<th>Ja</th>
<th>Nej</th>
<th>Vet ej</th>
</tr>
</thead>
</table>

Om ja, beskriv hur (kortfattat):

35. I vilken omfattning bidrar leverantörer till utvecklingen av företagets erbjudande?

<table>
<thead>
<tr>
<th>Betydelselös</th>
<th>Avgörande</th>
<th>Vet ej</th>
</tr>
</thead>
</table>

36. I vilken omfattning bidrar kunder till utvecklingen av företagets erbjudande?

<table>
<thead>
<tr>
<th>Betydelselös</th>
<th>Avgörande</th>
<th>Vet ej</th>
</tr>
</thead>
</table>

37. I vilken omfattning bidrar användare till utvecklingen av företagets erbjudande?

<table>
<thead>
<tr>
<th>Betydelselös</th>
<th>Avgörande</th>
<th>Vet ej</th>
</tr>
</thead>
</table>