Product Features in Commercial Open Source Software

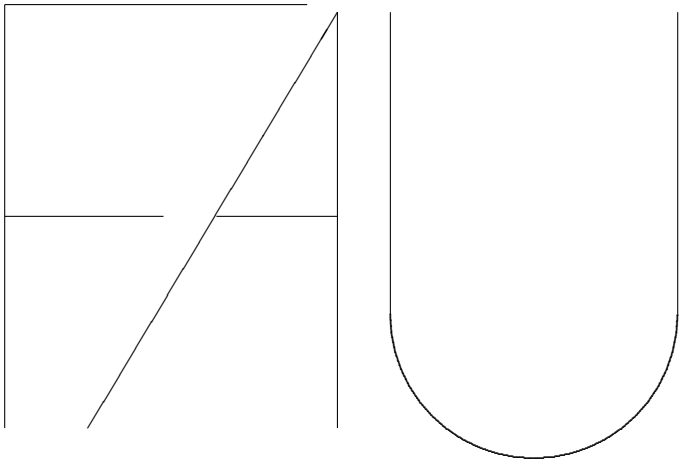
Bachelor Thesis

Florian Weikert

Open Source
Research Group

Department of
Computer Science
Faculty of Engineering

Friedrich-Alexander-
University of
Erlangen-Nuremberg



Abstract

Although open source software has developed from a negligible phenomenon to an integral part of today's business world, there are still companies which cannot see its commercial viability. As a consequence, these companies keep on ignoring this wide-spread trend, therefore missing possible profits and the chance to gain an advantage over their competitors. In order to provide arguments for the economic potential of open source software, a comprehensive model of commercially viable product features is described in the present thesis. This hierarchically structured model was created based on interview analysis, literature research and expert discussions and provides a detailed analysis of each feature, while explaining why these features are of economic value. Additionally, practical knowledge of open source companies is incorporated into this work through the use of a survey. The results of this survey are used to validate and expand the model and to generate insight about both importance and bundling of features. In the end, the reader will see which aspects of open source software products can be used to generate profit.

Contents

List of Figures

List of Tables

List of Abbreviations

AGPL GNU Affero General Public License

FSF Free Software Foundation

GPL GNU General Public License

IP Intellectual Property

OSD Open Source Definition

OSI Open Source Initiative

OSR Open Source Research Group

OSS Open Source Software

VAR Value-Added Reseller

1. Introduction

While traditional software companies try hard to keep their source code proprietary, so-called Open Source Software (OSS) describes a different approach. This term refers to "software which is [typically] available for free, whose source code is publicly available and which can be modified and re-distributed at no costs" [Wei11].

Several papers and studies indicate that open source software has become a substantial part of today's business world and therefore is a highly relevant topic [Rie07] [Rie11]. This is not only true for the sole use of such software – even more important, there are numerous companies creating profit based on open source software [KGS10]. For example, mobile phones running the open source operating system Android accounted for more than 52 percent of all smart phone sales in the third quarter of 2011 [PS11]. This trend seems to continue, as predictions state that "by 2012, 80 per cent of all commercial software will include elements of open-source technology" [PG08]. Moreover, the commercial value of OSS is demonstrated by the fact that the open source company MySQL AB was bought for one billion dollars [Mar08].

Despite this development, some people remain sceptic about the economic relevance of open source software. They cannot see the possibility of commercial exploitation, as expressed in the following quote: "How can you make money if you give the software away for free?" [Aar06, p. 299]. Consequently, these people keep their focus on traditional closed source software. As a result, they miss chances to create profit and might even face economical disadvantages against competitors which rely on open source software.

1.1. Contributions

This thesis shows that there are numerous possible features of open source products which have a financial value. Consequently, these features can be used to create a commercially viable product based on open source software. The major contributions of this thesis can be summarized as follows:

- The development of a hierarchically structured model of commercially viable features in open source software;
- A detailed discussion of each feature, including an analysis of possible dimensions and economical significance;

- An exemplary application of the model on the basis of three open source companies;
- The empirical evaluation of the existing model through the use of a survey;
- Conclusions regarding the bundling and ranking of product features as derived from this survey;

Originally, this thesis aimed solely at "single-vendor open source firms" [Rie11, p. 91]. This term describes companies which own all intellectual property rights to their software, thus being in complete control of their open source products. This type of company should be contrasted with distributors and service providers, since these firms usually capitalize on software created and owned by other companies. During the process of constructing the model, however, it became obvious that these three types have a lot of possible product features in common. As a result, we decided to consider them in the model as well. Chapter 6 provides an example of how this model includes all three types of companies.

1.2. Structure of this thesis

This thesis is structured in the following way: First of all, Chapter 2 introduces basic concepts and definitions which are of fundamental importance for understanding its content. An overview of related work in the domain of commercial open source software is presented in Chapter 3. Chapter 4 describes how the scientific process looked like, i.e. how the model was created and the empirical study was conducted. The result of this process, a model of product features in commercial open source companies, can be found in Chapter 5 where each of its features and their details are discussed in detail. An exemplary application of the model is shown in Chapter 6 where three concrete open source companies are classified based on the model. The empirical study evaluating the model and its results is subject of Chapter 7. Limitations to the content of this thesis are identified and discussed in Chapter 8. Finally, Chapter 9 provides a reflection of the present thesis by showing possibilities for future work, evaluating its major contributions and providing a concluding summary.

2. Basic concepts and definitions

This chapter introduces and defines several concepts which are being used throughout the thesis. For example, unique characteristics of open source software and their implications on open source companies are presented. Finally, three possible business models for such companies are discussed.

2.1. Open source software

The most crucial definition in this thesis is the one of *open source software*. As already mentioned in the beginning of Chapter 1, this term describes a special type of software with unique characteristics. The prevalent definition – the Open Source Definition (OSD) – was given by the Open Source Initiative (OSI) which can be found on its website. In regard to this thesis, two of these ten characteristics are of eminent importance: "availability of source code" and "free redistribution" [Opeb, para. 1-2]. These ideas are clearly opposed to closed source software where source code is not publicly available and redistribution is forbidden under most circumstances.

2.2. Intellectual property and licensing

Unlike physical goods, software remains a purely abstract concept which cannot be grasped or touched ("intangible good"). Additionally, software – as all intangible goods – has the advantage that it can be duplicated easily without requiring further effort or costs. Similar to music, software constitutes Intellectual Property (IP) rather than tangible property. There exist several ways to protect intellectual property, e.g. trademarks, patents and copyright. By default, intangible goods are usually the intellectual property of their creators, unless IP rights are explicitly transferred to another party [Hel11]. As described by Bonnabeau, people or organisations owning IP rights to a specific product have certain privileges, including the possibility of granting usage rights to others (*licensing*):

> "The right to use Software is typically given in the form of a license, rather than a sale. A sale would give the user all rights that are associated with ownership of the Software (e.g., sale or license of the Software to third parties, the right to copy and create derivative works). As vendors desire to retain such rights in the Software, the vendor conveys limited rights to the user in the form of a license" [Bon, p. 3].

Consequently, the party owning the software (*licensor*) permits others (*licensees*) to use this product by providing a *license*.

2.3. Open source licenses

An *open source license* is a software license which (a) is approved by the OSI and (b) does comply with the OSD (see Section 2.1) [Opea]. Exemplary licenses are the GNU General Public License (GPL)[1] and the Apache License.

The GPL license is the most prominent example for a special type of licenses, so-called "viral copyleft" [Hel11, p. 16] or "[strong] reciprocal" [Vö05, p. 118] licenses. If someone uses software covered by a reciprocal license within his own software, he has to offer his software under the same license. As a consequence, embedding the source of reciprocal open source components in a software products means that this product has to become OSS, too.

2.4. Commercial open source companies

In this thesis, the term *commercial open source company* describes a company, which generates revenue based on one or more software products covered by an open source license. However, not all of their products have to be open source. Possible benefits of being an open source company include improved feedback, higher software quality, more efficient distribution, faster development and reduced costs compared to closed source companies [Rie10b] [CM10].

There are two important points to notice:

(a) Since the software has to be covered by an approved open source license, it has to comply with the definition of OSS. This implies that an open source company has to create a surplus value based on a product which is typically available for free, i.e. its commercial offerings have to compete with its own free products. Consequently, if such a company fails at this task, potential customers will stick to the free version and refuse to pay for any commercial offerings. This problem was also addressed by Comino and Manenti:

> "There is always the risk that the free version cannibalizes the market" [CM10, p. 4].

(b) The underlying open source software does not need to be intellectual property of the firm which capitalizes on it. As an example, commercial open source companies may offer paid services for open source software developed and owned by other companies. This is alleviated by the fact that source code is available, therefore allowing other companies to get a better understanding of the software.

[1] It complies with the OSD although it was created by the Free Software Foundation (FSF).

If point (b) does not apply to a particular company, this means that this company has to offer more than one version of their software. The source code of at least one has to be publicly available in order to meet the criteria of open source software. This version will be called *basic version* or *free version* in this thesis. Additionally, one or more paid versions – referred to as *commercial versions* or *commercial offerings* – are offered to generate revenue.

The surplus value mentioned in (a) is the major object of investigation in this thesis and will be examined in Chapter 5.

2.5. Classification of open source companies

Speaking of *commercial open source companies* suggests that this term describes a homogeneous set of companies. However, during interview analysis it became obvious that this category unites companies with distinct business models. The following classification is motivated by Krishnamurthy [Kri03], therefore the present thesis does not consider companies which provide open source software in order to sell complementary hardware.

Software producers create and sell their own software, i.e. they own the intellectual property used for their commercial offerings[2]. Consequently, they are an example for single-vendor open source firms (see Chapter 1). They may accept code contributions from external volunteers, although these have to transfer their IP rights to the company or grant permissions in terms of licensing [Rie09]. Additionally, they can offer supplemental services for their software. Prominent software producers are MySQL and Jaspersoft.

Service providers differ from software producers by the fact that they provide services for a third party product which is not their intellectual property. While they may choose to create intellectual property by writing software *on demand*, service providers are not obligated to produce software in general (e.g. pure consulting companies). Concrete examples for this type of company are OSSCube and ThoughtWorks.

Distributors create value by integrating a set of independent open source components into one final product. According to [Rie11], these components are typically not their intellectual property. However, they own the final configuration. Well known distributors are Red Hat and SUSE.

Although both software producers and service providers can create software and offer services, there exists a clear distinction between them: Software producers provide services based on their own products while service providers focus on third party products[3]. This difference is highlighted by Krishnamurthy as he speaks of "third party service providers" [Kri03, p.

[2] External libraries used in their software can be an exception to this rule. However, producers must acquire appropriate licenses in order to include these libraries in their commercial products.

[3] As seen in Chapter 6, they may also provide services for the results of their own development services.

10]. Additionally, software producers create their own software and offer it without changes to multiple customers, while service providers create individual software solutions for their clients on-demand. Consequently, software written by software producers exists before interactions with customers take place, whereas service providers have to interact with customers first before the custom software solution can be implemented.

Furthermore, the business model of an open source company determines its position in the value creation chain. Software producers can be positioned at the beginning of the chain, followed by Value-Added Reseller (VAR) companies and distributors while end users are located at the end. As discussed in Chapter 5, this position may have an impact on product offerings, since resellers and end users follow different goals, therefore being likely to have different requirements concerning certain product features.

3. Related work

The increasing economic relevance of open source software has not gone unnoticed in science. Consequently, several papers and studies address this issue by providing details on its commercial viability. Although the examples presented here mainly focus on open source business models, possible product features can be derived from their contents. In addition to showing relevant contributions, both their similarities and differences to the content of this thesis are discussed.

3.1. A. van Aardt: "Business Models on Open Source Software"

Van Aardt identifies ten strategies to capitalize on open source software [Aar06]. The description of these strategies also includes commercially viable product features which can be summarized as follows:

Packaging: Companies can integrate several independent open source software products in one package, configure this package and offer it on a CD. Consequently, customers do not have to spend time on configuration.

"Proprietary software extensions": Add-ons and extensions to open source software are developed as traditional closed source components and are sold to customers.

Additional licenses: Since companies are allowed to publish their own software under multiple licenses (*dual-licensing*), they can sell commercial licenses which include more rights and permissions than an open source license does (e.g. maintenance).

Support services: Customers want to get support if problems arise. Companies can react to this requirement by providing support services for open source software.

Integration services: Companies integrate open source software with traditional software, therefore allowing their customers to use both types of software side-by-side. Similar to packaging, the customers' effort is reduced.

Hardware: Some companies sell a combination of hardware components and open source software running on these components. They generate money by selling the hardware, i.e. open source is used solely to boost sales.

Training services: Companies can charge for teaching users of open source software how to operate it in an efficient way.

Guides: In addition to training, companies may publish documents and articles on how to operate it efficiently.

3.2. B. Fitzgerald: "The Transformation of Open Source Software"

Fitzgerald also analyses business strategies for open source companies [Fit06]. However, he does not provide a detailed discussion of features when talking about "value-added service-enabling" and "loss-leader/market-creating". For example, he only mentions support, configured distributions, advanced functionality and "complementary software products" in terms of commercially viable features.

3.3. F. Hecker: "Setting Up Shop: The Business of Open-Source Software"

Similar to van Aardt, Hecker discusses possible open source business models [Hec99]. In regard to product features, he mentions services and deliverables which are also addressed by van Aardt, e.g. support, training, hardware components and the distribution of software on a CD. Furthermore, he identifies additional features:

Printed manual: Companies offer their product documentation in printed form, i.e. customers receive a physical book.

Branding: Customers pay for the right to use the brand and trademarks of the manufacturer of a specific open source product.

Custom development: Specific requirements of customers can be met by providing a customized software implementation process.

Consulting: Open source companies charge for consulting services.

Complementary online services: Companies offer paid online services which can be used in conjunction with the open source product.

3.4. S. Krishnamurthy: "An Analysis of Open Source Business Models"

As mentioned in Chapter 2, this thesis considers three major business models which are based on the work of Krishnamurthy [Kri03]. His description of these models allows to identify several commercially viable features of open source software:

"Distributor": First of all, distributors can provide their software on a CD. Secondly, they may also charge for services such as support, training and consulting. They can also sell regular software updates.

"Software producer": Companies in this category create closed source software based on an open source product and sell it to their customers.

"Third party service provider": This type of company offers services for an open source product created by another company or a non-profit community. The only explicit example given here is the one of support.

3.5. D. Riehle: "The Single-Vendor Commercial Open Source Business Model"

Riehle classifies product features into four revenue sources [Rie10b]. In addition to these categories, he identifies several individual features [Rie10a]:

"Core product": This category refers to *dual-licensing*, e.g. companies also sell their products covered by a *commercial license*. Consequently, customers can avoid the restrictions of an open source license while receiving more rights, e.g. *indemnification*.

"Whole product": Companies sell an advanced version of their open source product, e.g. software with increased functionality or additional *utilities* ("freemium").

"Operational comfort": Possible features in this category are services which help at operating the software as desired. Examples include "incident-based support", "24x7 hotline [support]" and "update service".

"Consulting services": Companies can charge for "training, documentation and implementation services."

3.6. Other papers

As seen in this chapter, the term dual-licensing appears in several papers. An in-depth discussion of how this model can be employed and why an additional commercial license is a reasonable product feature can be found in various papers [CM10] [VÖ03].

The current subject was also addressed in a previous paper [Wei11]. It follows a feature-oriented approach to commercial open source products. However, since its result were completely integrated into the theoretical part of this thesis, its details will be subject of Chapter 4 when the research process is described.

3.7. Comparison with this thesis

The focus of the previously mentioned papers lies more on business models and business strategies. Possible features are only used to illustrate these models while no further analysis of their dimensions and details takes place. The present thesis, however, aims at exactly this topic. Although Comino and Manenti discuss dual-licensing and commercial licenses in detail [CM10], the present thesis will do so for more than just one feature. Furthermore, it presents a comprehensive model which illustrates possible features in a hierarchical structure. Finally, results from an empirical study are used in order to evaluate this model and to generate new insights about further topics such as the importance of individual features and common patterns in their appearance.

4. Methodology

This chapter aims at providing insight on how the actual research process looked like. It can be divided into five major steps: First of all, we created the initial model. Secondly, three consecutive iterations of revising the model took place. The final model was then used to design a survey which should evaluate the theoretical work. Next, we published the survey and advertised it to a specific audience. Finally, the results were analysed by us.

4.1. Creation of the initial model

We developed the first version of the present model in a previous paper [Wei11]. It is based on a series of seven confidential interviews performed by Dirk Riehle in the years 2009 and 2010. At the time of the interviews, the five interview partners had at least 18 years of experience in the software industry and held different jobs, from software engineers to chief executive officers. They represented three open source companies, which were founded at least six years prior to the interviews and which were – and still are – established companies in their respective business segment. These companies comply with the definition of an open source software producer. Furthermore, all of them are located in the United States of America.

First of all, we analysed the transcriptions of the interviews in order to gain a better understanding of the domain of commercial open source software. We marked and collected text segments containing domain-specific information, e.g. "Pro edition differs from the community edition it does have some features that are not in the community edition and it's available under traditional commercial license" [RI10a]. Then we divided these segments into individual concepts and assigned codes to them. For example, the quote mentioned before led to four codes: *pro edition*, *community edition*, *different features* and *traditional commercial license*.

After we had analysed these interviews and encoded all relevant segments, we grouped related codes together. For instance, the codes *commercial license*, *warranty*, *indemnification*, *pro edition* and *enterprise edition* formed one group since they frequently occurred in conjunction with each other. Likewise, *GPL*, *open source license* and *community edition* represented another group.

Although the codes in each group were related to each other, no statements could be made on how these relations looked like. Therefore we started to structure each group. By doing so, we generated more abstract concepts described through the individual codes. One example for such a concept was named *commercial offerings* which originated from the first group of codes

mentioned before. We regarded the codes *pro edition* and *enterprise edition* as two exemplary instances of this concept while *commercial license* described one of its possible characteristics. Additionally, a second major concept, *non-commercial offerings*, could be identified.

Next, we focused on the relations between these two concepts by examining codes which appeared in conjunction with both of them. These codes stressed the differences between the two concepts, e.g. *functional differences*, *feature differences*, *advanced documentation* and *professional only extensions*. They implied not only a difference but also the superiority of the *commercial offerings*.

Consequently, we focused on these differences by finding exemplary codes, i.e. characteristics which constituted the superiority of the *commercial offerings*. For example, the previously mentioned codes of *commercial license*, *warranty* and *indemnification* but also others such as *support*, *training* and *documentation* were considered in this step. Similar to the previous process, we combined these codes in order to form new concepts. For instance, we assigned codes such as *24/7 support option*, *incident-based support* and *subscription-based support* to the category *support*. Additionally, these codes allowed to add dimensions to their respective concept. In the case of *support*, the dimensions included *availability*, *channel*, *support type* and *supported versions*. Individual codes allowed to assign possible values to these dimensions, e.g. *availability* could either be *12/5* or *24/7*.

After we had converted all codes into concepts and dimensions, we combined concepts into categories, whenever possible. For example, *functional features* and *non-functional features* formed the category *features*. As a result of this process, we turned the previously unsorted codes into a hierarchical model of concepts.

Next, we gathered more data from the product portfolios of the interview partners' companies in order to refine the model. In the case of *support*, this led to new dimensions such as *authorized contacts* and *number of incidents* Figure 4.1 shows the model as presented in the paper *How To Earn Money With Open Source Software*.

4.2. Iterative revision of the model

For the present thesis, we revised and extended the original model during three consecutive iterations. This process included analysis of literature, examination of product portfolios from more open source companies and discussions with an expert, namely Professor Dirk Riehle (see Table A.1 for a list of these companies). With each iteration, we added new product features while the dimensions of the existing ones were refined. Additionally, the hierarchical structure was improved, leading to the three top-level categories of *legal features*, *IP related features* and *service features* in the final version of the model. A detailed presentation of this model is subject of Chapter 5.

Figure 4.1.: Original version of the model as presented in a previous paper. Source: [Wei11]

4.3. Survey design

Since the theoretical work was conducted from a qualitative point of view, we tried to improve its significance by employing a quantitative empirical study. This study consisted of a survey designed with the following goals in mind:

- Evaluation and completion of the model presented in the first part of this thesis;
- Rating of product features based on number of occurrences and importance;
- Finding recurring patterns of how features are grouped together (*Bundling*);

As the survey required detailed knowledge about individual features of products, the ideal target audience were product managers. In order to accomplish these tasks, we structured the survey as follows: The first part consisted of an introduction, asking for the dominant business model and the number of products of the participants' companies. These questions were mandatory since they were required for the latter parts of the survey.

The next sections had a similar structure, with each of them referring to a specific category of features. As mentioned before, three main categories of features were considered in the final model (see Section 4.1). However, since the category of *service features* was very large, we divided it into four sub categories: *support*, *training*, *general services* and *client-specific services*. The other two main categories from the model were named *legal issues* and *proprietary*

intellectual property, therefore resulting in a total of six categories in the survey. For each of these categories, we added one section with the following structure:

The first question aimed at finding out whether specific product features were offered by the participants' companies as part of one or more products. This was implemented by using a matrix, with all possible features from the category on the y-axis and up to three products of the current participant's company on the x-axis. Participants should mark each combination of feature and product if the feature was part of the particular product. The data gathered from the first question allowed to make assumptions about bundling and the number of occurrences of individual features.

The second question asked for additional features which were missing from the survey. This question was included in order to extend the model by finding features which were not yet included in the model.

Finally, the last question asked the participants to rank the features in the current category from most important to least important. We formulated this question in a subjective way: "Judging by your experience, how important are the following features to your customers?". However, this was intended, since we wanted to find out whether there is a correlation between this subjective importance and the number of occurrences of a feature. The order of these questions and the categories was fixed, i.e. every participant had to answer the survey in the same sequence.

During the design phase, it became obvious that the complete hierarchical structure of the model could not be transferred to the survey. If all details and dimensions were included in the survey, it would have become too comprehensive, therefore possibly reducing the motivation of participants. Due to the fact that the survey left fewer place for explanations and definitions than the model did, terms and formulations had to be adjusted. A printed version of the entire survey can be found in Chapter B in the appendix.

4.4. Publication of the survey

In order to reach an international audience, we decided to create an electronic survey in English. Therefore the survey was implemented using the LimeSurvey software.

Prior to releasing the survey to the public, a first version was issued to Professor Dr. Dirk Riehle, one of his Ph.D. students and two computer science students of the University of Erlangen-Nuremberg. Their feedback led to improvements in terms of formulation. Due to limited time, however, we did not conduct any formal tests for reliability, objectivity and validity.

Next, we published links to the survey in various ways: First of all, two German open source newsletter were used to advertise this survey. In order to reach an international audience, we posted a blog entry on the web site of the Open Source Research Group (OSR) which was

further promoted through messages on Twitter. Finally, selected industry contacts of Dirk Riehle received an email asking for their participation in the survey.

4.5. Analysis of survey results

First of all, invalid responses had to be filtered out prior to data analysis. In order to achieve this, we used a statistics software called R. Additionally, we defined valid as "the number of products has to be lower than 50 and at least one question after the introduction has to be answered".[1] Furthermore, we removed one more answer as its business model "research tool" did not make sense in our eyes.

As a result, 15 data entries were left, each of them containing information about up to three different products. Next, we created a new table with one entry per product and one column for each feature. Possible values were zero, one or "not a number" if the question was not answered. The resulting table showed the mapping between all 27 products and 34 possible features.

Consequently, the frequency of a individual product equalled to its column sum divided by the number of products. This data allowed us to rank the features by their frequency (both for each category individually and in total).

In addition to this *implicit ranking*, the survey contained questions asking participants for an *explicit ranking*. For each category, participants had to rank the respective features from one (most important) to n (least important) where n was the number of features in this category. Consequently, we computed the results for each category individually. When evaluating these questions, we assumed a linear scale, from n points for the most important feature to one point for the least important one. Additionally, incomplete rankings were considered as well. Next, we divided the score of each feature by the maximum possible score of the particular category. This maximum score equalled the number of features multiplied with the number of responses for the category.

We then used the table containing all 27 products in order to detect common bundles of features. However, we decided to consider products from complete responses only. This decision was motivated by the fact that few participants answered the latter questions of the survey while the categories at the beginning received more answers. Consequently, considering all responses would have corrupted the results. As a result, we used only 16 products to find bundles.

Since there were 34 features in total, the size of potential bundles ranged from two to 34. For each value of this size, we calculated all possible feature combinations (based on the previously mentioned table). When such a combination occurred in more than 50 percent of all 16 products, it was recorded. The resulting combinations were then sorted by the number

[1] The number of 50 was chosen since it was the first value from the answers which seemed to be unrealistic.

of occurrences, as seen in Chapter 7. Next, we removed redundant entries, i.e. bundles which were a strict subset of larger bundles but which had the same number of occurrences.

Finally, we transformed these results into diagrams and tables. The R programs written for analysis and presentation can be found on the enclosed CD.

5. Model of product features

This chapter displays a comprehensive model of possible product features in commercial open source software. These features can be grouped together in three major categories:

1. Legal features
2. Features related to intellectual property
3. Service features

These categories are also subject of Figure 5.1. The following sections focus on discussing the features of each category while showing possible sub categories and providing more information on the individual features. However, an emphasis will be placed on legal aspects as they are not a familiar topic for someone working in the IT domain. For each feature, three different aspects will be looked at:

(a) How is the feature defined?

(b) Why are customers willing to pay money for it?

(c) What are its details and how do possible dimensions look like?

As illustrated in Section 2.4, commercial offerings of open source companies have to compete with versions which are available for free. Consequently, they have to create incentives for customers to actually spend money on the commercial products instead of using the free version. Therefore aspect (b) is of importance. The reader should keep this fact in mind when reading the following sections.

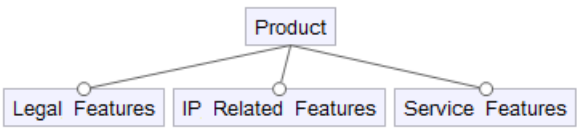


Figure 5.1.: Hierarchical breakdown of a product into three major categories of features

5.1. Legal features

The fact that open source licenses are different from traditional licenses was already covered in Section 2.3. As a consequence, the use of OSS may have legal implications which are not encountered when using traditional closed source software. Therefore open source companies may offer specific features addressing such implications. These features will be discussed in the following sections.

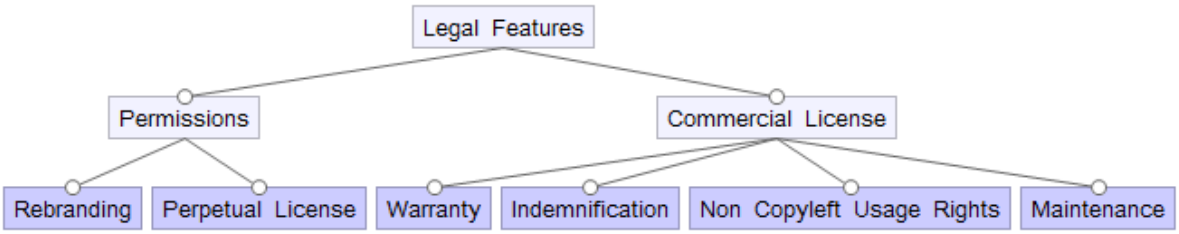


Figure 5.2.: Hierarchical structure of legal features. (Sub-) Categories are represented by light coloured boxed while dark coloured boxed stand for features.

5.1.1. Commercial license

As described in the previous sections of this thesis, software has to be offered under an approved open source license in order to be considered as open source software. However, additional commercial offerings based on such open source software may be covered by another license. This type of license was named *commercial license* and consists of four features, which will be discussed in the following.

5.1.1.1. Non copyleft usage rights

This feature aims at OSS covered by *reciprocal* or *copyleft* open source licenses (see Section 2.3). If customers included such software into their products, they would have to make their own software available under the same license. This would imply that their software became open source and its source code had to be publicly available as well.

However, this may not be acceptable to customers for various reasons. First of all, having to publish all source code means that customers would reveal their own intellectual property to others – including their competitors. Additionally, other risks arise if the source code is publicly available. If intellectual property like patents or copyright was infringed during development, this would lead to patent claims and lawsuits, thus resulting in additional costs and bad reputation for the company.

Consequently, the use of third-party open source software in closed source products is not trivial as risks and negative ramifications have to be taken into consideration. A detailed

analysis of such risks and possible approaches to deal with them can be found in chapters eight and nine of another thesis [Hel11].

As a result, it would be of value to customers if they could use the desired products covered by a *commercial license* without copyleft clause. Open source companies may satisfy this need by using a so-called *dual licensing* strategy, i.e. by offering the same piece of software under multiple licenses with different conditions [VÖ5]. This is only possible if these companies own the intellectual property they capitalize on.

5.1.1.2. Warranty

Warranty describes "a usually written guarantee of the integrity of a product and of the maker's responsibility for the repair or replacement of defective parts" [MWb].

According to Laurent, this guarantee implies that products "will perform a particular function" (*express warranty*) and "are suitable for use as generally intended" (warranty of merchantability *or* implied warranty). Furthermore, if the vendor declares that a particular product can be used for a specific task, he has to guarantee this as well (*warranty of fitness for a particular purpose*). Finally, *warranty against infringement* is another type of warranty dedicated to products constituting intellectual property, e.g. software: "Such a warranty is a guarantee by the seller [...] that the work that she is selling is in fact a work she has copyright to, generally because she is the creator of the work." If these guarantees are violated by a product, "the buyer has a remedy against the seller, generally either to have the price of purchase returned or to receive an equivalent but functioning item in exchange for the defective one", i.e. he has to compensate for "direct damages"[1].

However, open source licenses usually do not provide warranty. For example, the GPL license enforces no warranty at all, therefore transferring all risks to the users:

> "THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM 'AS IS' WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION" [Fre07b, para. 16].

From an economic perspective, there are two major reasons for this behaviour: First of all, the success of open source is based on the voluntarily contributions of programmers. However,

[1] All quotes in this paragraph were taken from the work of Laurent [Lau04, p. 11-12].

as expressed by Välimäki, warranty clauses are likely to constrain these contributions: "Requiring a certain kind of minimal warranty would [...] frighten developers who can't accept any kind of liability for what they do" [VÖ5, p. 115].

Secondly, a warranty clause implies additional risks and therefore additional expenses for the software vendor while he does not earn any money with this particular product in return. These risks are especially high if the company accepts contributions from the community, i.e. virtually everybody can participate in the development process by writing source code. Providing warranty for such software puts even more risk at vendors since they can be held responsible for any errors caused by community written code. The only solution for this problem would be the introduction of an intense testing and approval process of all code, thus resulting in even more costs.

Nevertheless, customers are interested in having a product covered by warranty since it reduces the risks associated with their bargain. Consequently, open source companies may offer warranty as a part of their commercial licenses and charge for this. However, several aspects have to be taken into consideration [DS07] [Bon]:

Subject to warranty: First and most important is the description of which characteristics and functions are covered by warranty. For example, if companies include third party software in their products, it might be reasonable to offer warranty only for their own components instead of the whole software.

Warranty period: Warranty is often limited to a certain period after the purchase, therefore both start and end of this period have to be announced.

Actions in case of remedy: Companies can specify which actions have to be taken upon warranty violations. Possible actions include repairing the defective product, providing a replacement or refunding its price.

Covered damages: It is important to know whether the warranty clause also covers "consequential damages" [Lau04, p. 12], i.e. damages to people or the environment which were caused by a defective product.

Limitations: The vendor may limit the validity of the warranty clause to specific situations, i.e. certain preconditions have to be met before his customers may hold him liable. Additionally, customers' negligence may be considered as well. For example, if any defects occurred while a customer operated the product in an inappropriate way, warranty might not apply.

By incorporating these aspects, open source companies may even offer multiple contracts with different warranty clauses, thus enabling them to provide a scalable model of warranty.

One contract might include a short warranty period with a lot of limitations at a low price, while others might include longer warranty periods and less restrictions at a higher price.

5.1.1.3. Indemnification

Similar to *warranty*, *indemnification* poses an important legal concept when it comes to economic transactions. Classen defines this concept as follows:

> "General indemnification clauses usually address the liability of one party to the other for liability the first party incurred to a third party as a result of the second party's actions. [...] Indemnification clauses are common means to contractually allocate risk between the parties" [Cla07, p. 55].

For example, if a vendor sells products from another company without an existing indemnification agreement, he would be responsible for covering possible damages inflicted by these products to his customers. If an appropriate indemnification clause existed, the company would have to cover these damages, therefore relieving the vendor. Consequently, the vendor is interested in having such an indemnification agreement in order to avoid costs and legal consequences.

This holds true for customers of open source companies, as they can avoid the responsibility for damages which resulted from the use of open source software. Similar to warranty, however, open source companies do not want to provide indemnification for their non-commercial offerings since it puts additional risks and costs on them. Nevertheless, they can provide indemnification as a commercial feature.

Several possible limitations to an indemnification clause have to be considered [Cla07]:

- It is important to know which types of damages are covered. A possible classification distinguishes between personal damages (to humans), property damages and infringement of IP rights. Consequently, indemnification may cover only some of these types. Furthermore, the licensor's liabilities may be further narrowed down by considering only specific aspects, e.g. only patent violations are relevant rather than all IP infringements. Secondly, indemnification clauses can either be limited to direct damages or include consequential damages as well (see *warranty*). Finally, it can be relevant whether the damage is claimed by a third party or the licensee himself.

- Indemnification agreements can declare an upper limit for possible financial compensations. Additionally, they may limit the actions of the licensee, e.g. by allowing only repairs instead of refunding.

- Claims may only be considered from specific prosecutors.

- It has to be declared what happens if the licensee causes damages through his own negligence.
- Specific actions enabling indemnification can be included in the agreement.

Furthermore, the agreement must explicitly state how liabilities are allocated between licensor and licensee. For example, the licensee may be obligated to pass all claims to the licensor and support his defence [Cla07]. Additionally, the agreement has to specify which actions have to be undertaken by the licensor if indemnification is necessary. Such actions include the removal of questionable or defective parts, the acquisition of missing licenses and the financial compensation of the licensee. In the latter case, it must be specified whether the whole price or just parts of it are refunded to the licensee.

5.1.1.4. Maintenance

Maintenance can be defined as follows:

> "(1) The process of modifying a software system or component after delivery to correct faults, improve performance or other attributes, or adapt to a changed environment. [...]
> (2) The process of retaining a hardware system or component in, or restoring it to, a state in which it can perform its required functions.[...]"

(Source: *IEEE Standard Glossary of Software Engineering Terminology*, IEEE Standard 610.12-1990, p. 46)

In this thesis, two terms characterize such changes to the software in order to reflect the different goals of maintenance. We define them as follows: *Updates* are modifications for improving performance, implementing new requirements and adapting the system to changes. Modifications with the intention of fixing faulty behaviour are called *patches*.

As stated by Erlikh [Erl00] (see [Kos04]), more than 90 % of the total costs of a particular software are spent on its maintenance. Consequently, maintenance is a relevant topic for software consumers and producers alike, regardless whether in closed or open source. According to Kim, Djamaludin and Murthy, *maintenance* is related to *warranty* as software companies are constrained to maintain their products during the warranty period [KDM04]. However, Section 5.1.1.2 shows that open source companies usually provide their free products "as is", i.e. there is no warranty. As a result, these companies are not obligated to offer maintenance by fixing faults in the software and providing updates. Even though they are interested in keeping their software as bug-free and feature-rich as possible, non-paying users have no right to demand that these updates are delivered to them immediately as well as regularly. Therefore these users are forced to perform maintenance by themselves or to rely on the open source

community, if possible. This might not be acceptable as it involves either more effort and increased costs or – in the later case – is not reliable.

Open source companies can provide a solution by offering maintenance as part of their commercial offerings, thereby creating another source of revenue while satisfying the customers' needs.

5.1.2. Permissions

As described in Section 2.5, open source software is not only employed by end users. Furthermore, other companies may use such software as part of their own products (*service providers*, *distributors*). However, as end users and resellers use the software with different intentions, their individual requirements for the product will differ as well.

5.1.2.1. Rebranding

One example for different requirements is the handling of trademarks and logos. Companies like Alfresco and Openbravo enforce that their trademarks must not be removed from their open source software. Additionally, they request that redistributions of their software – modified or not – contains a visible notice, stating that parts of it are their respective intellectual property. Furthermore, resellers must also point out that they are "not endorsed by or affiliated with" these companies [Alf10] [Ope08]. End users are likely to not care about these regulations, as they do not affect their user experience. However, resellers may feel different about this issue. In regards to marketing and customer perception, it is beneficial for them to remove all external trademarks from the final products, therefore highlighting their own brand. Open source companies may react to this requirement by including a specific *rebranding permission* in their commercial licenses, allowing their customers to replace all trademarks.

5.1.2.2. Perpetual license

Another example for commercial permissions concerns the duration of contracts and its implications on usage rights. Similar to closed source software, open source companies can offer their products for an upfront license fee, allowing their clients to use the software for an unlimited period of time. However, companies like SugarCRM rely on a subscription-based payment model, i.e. their customers can use the software only as long as they pay for it [Sugg]. Consequently, these customers have to stop using the software when the subscription ends.

This might lead to problems, as the following scenario demonstrates: An open source company *OS* offers its commercial software product based on annual subscriptions. A VAR *V* wants to integrate it in his own products, therefore paying the required subscription fees to *OS*. Consequently, he is allowed to sell the software owned by *OS* as part of his products to his clients, thus effectively redistributing the software of *OS*. However, if *V* decided to cancel his

subscription, his end users would be affected by this decision as well. They would be forced to stop using the software[2], even though they might have signed a contract with *V* which guaranteed them unlimited time of usage. In order to solve this problem, *V* would have to continue paying subscription fees to *OS* until the last one of his customers stopped using his product.

OS can offer a solution for this dilemma by permitting *V* to offer a *perpetual license* to his clients. Consequently, they might continue using the software even though *V* has cancelled his subscription. Furthermore, *OS* can charge for this permission.

5.1.3. Managed release cycles

The term *managed release cycles* describes that new versions of a software product are released on a regular basis with a time span of at least several months between two consecutive releases. Furthermore, the vendor spends time on quality assurance and testing, making sure that these software releases are stable.

As discussed by Feller [FF00], open source projects have a high release velocity, i.e. new versions are released frequently ("more than one release per month" [Hel11, p. 8]). According to Feller and Fitzgerald, this distinguishes open source projects from traditional closed source projects:

> "OSS projects are generally characterized by *rapid, incremental release schedules*, in which limited extra functionality is added in each release. Proprietary software companies tend to follow the opposite model, introducing substantial changes in software products but very infrequently" [FF02, p. 24].

On the one hand, this has several advantages, as pointed out by Rossi [RRS09]: Bugs are fixed fast and new feature requests are handled early. On the other hand, however, such frequent releases may lead to several problems.

First of all, frequent changes in the software require frequent adoptions in its operating environment, thus leading to increased effort for the customers. This may not be acceptable to them, as described in one of the interviews: "Once we're running a production system, you really don't want to have to upgrade and modify it too many times there" [RI09]. This quote originates from one of the interview partners who also suggested a release frequency of two or three releases per year.

Secondly, less time between two consecutive releases implies that less time can be spend on quality assurance and testing[3] This may lead to unstable and faulty software.

According to Weinstock and Hissam, frequent releases may also have a negative impact on software security [WH05]. When both number and frequency of changes to the software

[2] Or at least the components which belong to *OS*.

[3] Although one has to admit that changes in open source releases are typically smaller, therefore requiring less testing.

increase, is it difficult to maintain a stable state of the software. Therefore it is getting harder to detect whether these changes lead to new weak points in the software. As a result, the software may be vulnerable to attacks from the outside, e.g. *denial-of-service* attacks. Consequently, customers might be interested in having less frequent releases, leading to reduced effort, improved stability and higher quality.

5.2. Intellectual property related features

This category refers to features which constitute intellectual property. An hierarchical overview is provided in Figure 5.3.

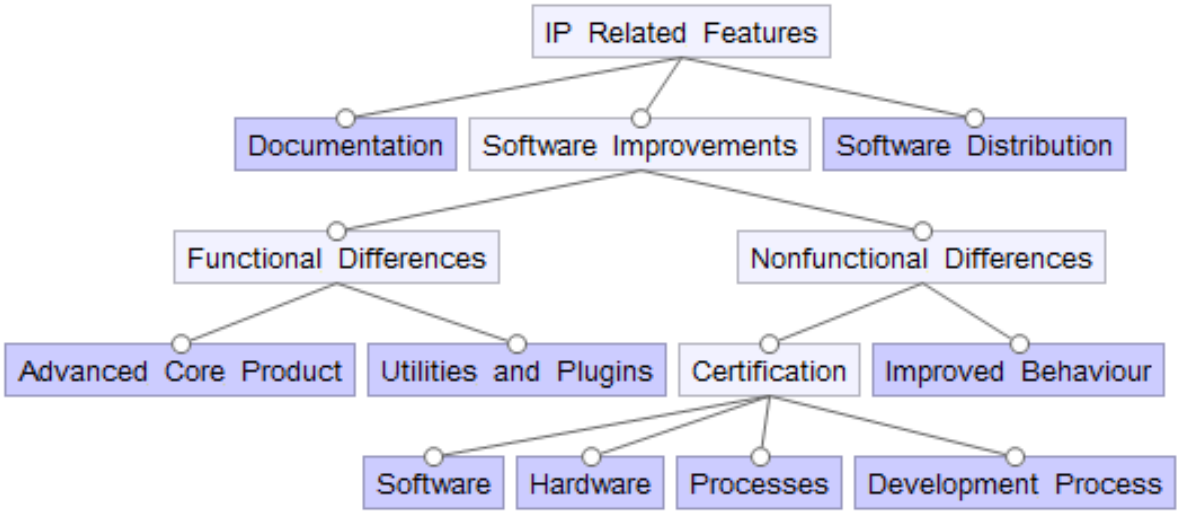


Figure 5.3.: Hierarchical structure of IP related features. (Sub-) Categories are represented by light coloured boxed while dark coloured boxed stand for features.

5.2.1. Software improvements

In the context of software companies, software is obviously the most important manifestation of intellectual property. Besides from providing software whose source code is freely available, companies can offer additional proprietary software and charge for it. However, this software must offer a surplus value compared to the first version. This category was termed "advanced features" in a previous paper [Wei11] and can be categorized in the following ways:

Functional differences exist when the commercial products either "offer a greater range of functions" [Wei11, p. 6] or provide different functions than the free version, i.e. they can "do more" or "do different things". These differences can be either implemented in *utilities and plug ins* or in an *advanced core product*. The first one means that new functions are realized through a separate piece of software which can be used in conjunction with the free version,

offering complementary services. An *advanced core product* can be defined as the union of the entire free product and additional features, i.e. the code of the free version is a strict subset of the code of the commercial product.

On the other hand, *non-functional differences* are present when both free and commercial versions provide the same functions, but the commercial products can execute them more efficiently, i.e. they can "do the same things better". One example is *improved behaviour* which consists of the following aspects:

Scalability: The free version may be limited in its supports for distributed and parallel computing. For example, it can be implemented to use not more than a specific number of CPU cores. This limitation can be removed from the commercial version, therefore allowing the application to scale with each new processor or computer being added to the system or network.

Performance: The commercial versions may include optimized algorithms, thus allowing a faster execution of their tasks. Similar to scalability, the use of resources which are critical to performance may be artificially limited in the free version, thus giving an advantage to the commercial offerings.

Security and safety: Improved mechanisms for preventing external attacks and internal failures may be implemented in the commercial products. Examples include better authentication mechanisms, improved encryption and automated backup of critical data.

Availability and reliability: A strict process of testing and quality assurance can be applied to the commercial products in order to ensure stability and accuracy, thus increasing the degree of reliance.

Comfort: Functions in the commercial version may be operated in a more convenient way than in the free version. One example is LimeSurvey, a software product for creating online surveys. This software offers a function for updating itself – either manually by downloading the files from the internet or by using the "3-Click ComfortUpdate" which takes care of the entire process. However, as of today, there are plans that this comfort feature will be exclusive to a commercial offering in the future: "[...] at some point in the future users might have to pay a small fee to purchase an upgrade key to continue using the feature" [Cle11]. Another possible comfort feature is the implementation of a better user interface in the commercial product.

Although it is not directly related to the software itself, the distribution process can be made more comfortable as well. According to Krishnamurthy, distributing the software on a physical medium such as CD or DVD might be reasonably since "most people are not comfortable with downloading the product from a web site" [Kri03, p. 6]. However, as his paper is from 2003

and the internet has considerably evolved since then, we believe that this feature is not of relevance in today's world.

Certification is also part of *non-functional differences*. Certified products are produced and tested in such a way that they meet specific criteria.

Customers usually employ one piece of software in a specific environment, i.e. by using it in conjunction with specific third party software, running on a certain operating system on a specific hardware platform. Furthermore, this software is employed in specific processes to met certain goals, which might put constraints on the function of the software and even on its development process. For example, some clients may request that the development process meets the requirements of the ISO/IEC 15504 standard (*SPICE*).

Due to economic and safety reasons, customers are interested in buying software which is known for meeting these requirements. Companies can satisfy this demand by certifying their software against popular software, hardware and processes. Since ensuring such compatibility requires additional effort, open source companies may charge for this feature.

5.2.2. Software distribution

In this case, the term "distribution" describes the final software product which is to be distributed rather than the process of distribution. Moreover, it can be defined as the integrated aggregation of individual and mostly independent software components from usually multiple vendors. "Integrated" means that these components are configured in a way that they work together. Therefore a distribution is more than just the sum of its parts. Additionally, the distributor may apply unique modifications to this software package. Prime examples for such software distributions are well-known Linux based operating systems such as Ubuntu, Debian and Fedora.

Although the individual components may be intellectual property of other companies and individuals, the results of aggregation, configuration and modification are intellectual property of the distributor [Rie11]. Consequently, single components may be used without the distributor's agreement while the package as a whole can only be used with his permission. Furthermore, the distribution is covered by a license (such as GPL) and the distributor may charge for its use.

One of the interview partners gave a reason why customers are interested in paying for such a distribution: "[...] it's one of the common problems with being an open source consumer and customer is that you don't kind of get this integrated solution easily right" [RI10a]. Therefore pre-configured *software distributions* can be employed to save time and to avoid integration efforts.

5.2.3. Documentation

Documentation refers to documents containing information about the structure and the usage of software. Consequently, these documents are an important resource for employees if they want to know how to use specific software. When the software is sufficiently complex, it can hardly be operated without any documentation. Furthermore, maintenance is only possible with an up-to-date documentation. As a result, companies are interested in buying software which is well documented.

Open source companies may generate money out of documentation by either releasing no or limited documentation for their free products while complete documentation is available for the commercial offerings.

According to Sommerville, there are certain types of documentation which are of relevance to software users: “Functional description”, “installation document”, “introductory manual”, “reference manual” and “system administratorsguide” [Som05, p. 5]. In practice, an open source company may only release one type of documentation for free while the rest has to be bought. Besides from the amount of documentation, its quality can be used to differentiate as well. Commercial documentation can include qualitative improvements, such as an index and practical examples.

Furthermore, open source companies may publish some of their documentation in printed form, as some clients might prefer to have a physical copy of the reference manual rather than just an electronic version.

5.3. Service features

This section covers the last one of the three top-level categories. As displayed in Figure 5.4, four major sub categories can be identified: support, training, general services and client-specific services.

5.3.1. Support

In terms of information technology, *support* is “assistance provided by a company to users of its products” [MWa]. If a non-trivial problem concerning the software or its operation arises, customers will want to contact support in order get immediate and reliable help.

However, commercial open source companies do not support their free versions since there is no financial benefit to them. Consequently, users have to rely on community support, i.e. by posting on public forums and mailing lists [Hel11, p. 47]. Since this kind of support is provided by volunteers, it is not reliable in terms of correctness and response time. As a result, costumers want to buy professional support offered by a company.

When looking at support, the following dimensions are of importance:

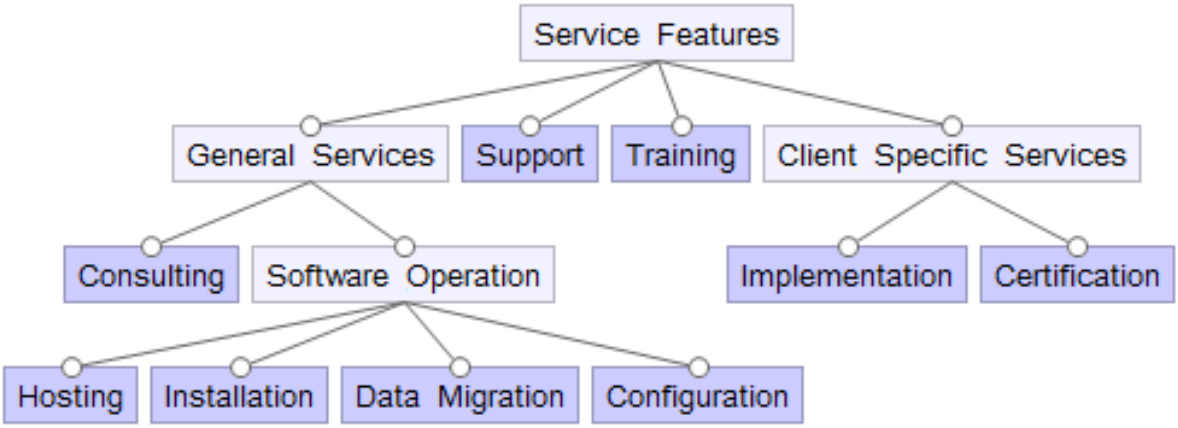


Figure 5.4.: Hierarchical structure of service features. (Sub-) Categories are represented by light coloured boxed while dark coloured boxed stand for features.

Type and channel: This dimension describes how customers can get support and whether they actively engage with support personnel. One possibility is *managed support*, i.e. customers receive help from employees of the company and may interact with them. Possible channels include phone, email, online chat[4] and forums maintained by the company. Additionally, companies may offer on-site support by sending employees or external experts to their customers, if practicable.

The other possibility is called *unguided support*. Rather than interacting with other people, customers are provided with resources which help them to solve their problems by themselves. For example, companies can provide an online knowledge base, i.e. special documentation aiming at solving typical support problems and FAQs.

Quality: When a problem arises, customers typically expect an immediate solution. To them, two quality metrics are of importance: First, the *response time* has to be low, i.e. the time between issuing and solving a problem. Second, companies who work 24/7 may request that support for their products has a similar *availability*. As discussed in a previous paper, availability has an impact on response time "since a weekday-only support implies long response times over weekends" [Wei11, p. 7]. Additionally, a dedicated support representative may be offered, making sure that all requests of a single company are handled by the same person.

Quantity: This dimension refers to quantitative characteristics of support. For example, the number of support incidents can be limited, although this applies mostly for subscription based support. Alternatively, companies may offer support on a "pay-as-you-go" basis by charging for each incident individually without requiring a contract. Furthermore, it is important how many of the customers' employees are allowed to contact support ("authorized contacts").

[4]This can either be instant messaging, web-based chats or chat rooms such as IRC.

These dimensions can be used to create multiple support offerings, each with different features and price. For example, open source companies may provide 24/7 support with guaranteed response time for demanding customers at a high price, while a support option with a 12/5 availability and longer response time can be offered at a lower price.

Support also includes requests by customers to fix a specified bug, therefore overlapping with the concept of *maintenance*. In the present thesis, however, maintenance is seen as an implication of warranty, while support constitutes a separate service feature which can be bought after warranty has ended. Nevertheless, these concepts are hard to distinguish in practice since customers will contact the company in similar ways, regardless whether the product is subject to warranty or support service was bought. Therefore the dimensions mentioned before also apply to the practical implementation of maintenance.

5.3.2. Training

Open source companies may offer *training* to their customers, i.e. employees of the customers are taught how to operate specific open source software in an efficient way. Similar to closed source software, training is an important offering to customers, since qualification and knowledge of their employees is crucial to their economic success.

Training can either take place online via a video stream (*webinar*) or offline in a classroom. Additionally, it can be differentiated between *self-study* and *instructor-led* training. In the first case, only documents and other resources are provided while in the second case an interaction with a person is possible. An offline training in a classroom is always led by an instructor. Furthermore, companies can offer a *professional certification training*. Upon the successful completion of this training, attendants will receive a certificate attesting that they have gained specific knowledge in regards to the subject of the training. This is especially valuable for employees who want to prove their qualification.

5.3.3. Client-specific services

This category describes services which are particularly tailored to fit the individual needs of single customers. One possible service is *custom implementation*. This can either mean changes to the implementation of an existing open source product or the complete development of a new, customized software.

The first case aims at customers whose needs cannot be fully satisfied by the original software. For example, they might need extra interfaces to other software products (e.g. SAP) or specific additional features. A software vendor can meet these needs by providing "on demand" implementation of software, e.g. by writing additional interfaces or implementing new features.

This creates several advantages for the software company: First of all, its customers can use the software even if the original version does not fulfil all of their requirements. Consequently,

customers are more likely to keep on using this software instead of buying from another company. Secondly, the software vendor has another service to charge for. In addition to that, he also has the possibility to include those special implementations in future releases of the original software version if more customers demand the same features. This leads to even more revenue and better quality due to the feedback received by customers.

The second case is useful for customers who have very special requirements not supported by existing software products.

Custom certification is another possible service. In order to keep expenses low, software vendors are likely to certify their software only against a selection of widely used software, operating systems and platforms. However, some clients may use very specific soft- or hardware for which no certification is offered. To solve this problem, software vendors might offer a *custom certification* service which can be bought by such clients.

Although *custom implementation* results in source code which is intellectual property, this feature is not listed in the category of IP related features. This distinction is intentional as this particular feature refers to a unique, made-to-order service while "normal" software is prefabricated and sold to multiple customers. The same applies for *custom certification*.

5.3.4. General services

Service features which are not part of the previous categories are covered here. They can be divided into *consulting* and other services related to the operation of the software.

5.3.4.1. Consulting

One possible service in this category is *consulting*. In addition to traditional consulting companies, open source companies may also offer consulting aiming at characteristics which are idiosyncratic to open source. For example, these companies may provide information and best practices for handling specific risks and potential dangers caused by the use of open source software (see Section 5.1.1.1). According to Josh Learner and Jean Tirole, IBM "has made open source software into a major focus for its consulting business" [LT05, p. 12]. Although IBM is not considered an open source company by the public[5], this quote shows that there seems to be a market demand for open source specific consulting services.

5.3.4.2. Software operation

The category *software operation* unites services which aim at supporting customers during the operation of the software. While the first three services normally take place at the customers' locations, *hosting* describes a different approach.

[5]Even though its open source consulting can be seen as an open source service provider, as defined in Section 2.5

Data migration: When replacing an old legacy system with open source software, customers will want to continue using their old data as it contains knowledge about the application domain. However, the format of this data can be different to the one used in the new software. Consequently, open source companies may provide a service which extracts the data from legacy systems, converts them into a new format and imports them into the new systems.

Installation: Open source companies can undertake the installation process of open source software, either online or by sending technical experts to customers. This is reasonably when this process is complicated or comprehensive.

Configuration: Especially complex systems require a lot of configuration before they can be operated. For example, settings have to be adjusted to reflect the specific operation environment. This can also be carried out by the vendor since his knowledge of the system allows a more efficient configuration.

Hosting: Instead of delivering the software to their customers, open source companies can operate the entire software product or just parts of it (e.g. data storage) in their own data centres ("Software-as-a-Service") while customers have remote access to the software via internet. This can be scaled down by providing only the physical hardware ("Infrastructure-as-a-Service") or the hardware in conjunction with a basic operating environment ("Platform-as-a-Service"). Similar to closed source cloud computing, customers can reduce their own IT infrastructure, therefore decreasing costs. Furthermore, they have less administrative effort as the cloud providers take care of their own data centres. Consequently, hosting incorporates installation and configuration as these tasks take place in these data centres as well.

Cloud computing in open source was explicitly addressed by the introduction of the GNU Affero General Public License (AGPL) license [Fre07a]. This license is almost identical to the third version of the GPL, except that it fixes a loophole regarding the redistribution of source code for hosted software. In addition to that, one of the interview partners pointed out that especially open source companies which do not cover their products by a reciprocal license could benefit from *hosting*: "A lot of people are realizing that particularly if you have a permissive license that hosting is one of the best ways to get to a commercialization strategy because you take the license issue off the table" [RI10c].

5.4. Annotations

Since the model is based on a qualitative data source, it is not possible to derive answers for all research questions of this thesis. For example, one cannot tell which features are most important since statements on this topic are subjective and not representative. One of the

interview partners identified indemnification as the most important commercial aspect [RI09], while another one claims that it is increased software functionality [RI10b].

Furthermore, service providers and distributors are excluded since all interview partners worked at software producers. As a result, a correlation between business model and offered features cannot be observed from the model.

Moreover, this models allows no conclusions on how these individual features are bundled together. Although it is noticeable that *warranty* exclusively appears in conjunction with *indemnification* in the interviews, one cannot generalize from this fact.

Consequently, a quantitative approach is needed in order to answer all research questions. Chapter 7 illustrates how an empirical study was employed for this purpose.

When comparing this model to traditional closed source software companies, it is noticeable that *non-copyleft usage rights* are the only feature which is exclusive to OSS. However, the perspective on some of the other features may differ as well. For example, closed-source companies are likely to see *warranty* as a nuisance since it implies additional expenses. On the contrary, open source companies regard such features as a possible way to generate revenue.

6. Classification of concrete open source companies

This chapter illustrates a practical application of the model described in Chapter 5 by portraying three concrete open source companies and analysing how their product features look like. Each of these companies represents one of the three business models described in Section 2.5.

6.1. SugarCRM, Inc. (Software producer)

SugarCRM, Inc.[1] produces a customer relationship management software product with the same name and provides services for it. Consequently, this company can be considered a software producer. The software is offered in four different commercial versions, with Sugar Ultimate being the one with the most features and the highest price. This version will be used as reference for this section [Suge]. If we mention "paying customers", we refer to customers of this version. Additionally, SugarCRM provides its fifth product version, Sugar Community Edition, under the reciprocal AGPL license, therefore meeting the requirements of an open source company [Sugl].

6.1.1. Legal features

Legal features were derived from the SugarCRM "Master Subscription Agreement" [Sugc]. Since SugarCRM sells its own software, it has the right to define the terms of the license.

Commercial license: SugarCRM provides its commercial products under a commercial license with *non copyleft usage rights*. In terms of *warranty* and *indemnification*, SugarCRM assumes minimal liability. The company only warrants that their services comply with the standards of similar services and that the software meets the specifications of the online user guide. If this does not hold true, its customers may cancel their subscriptions and get prepaid fees refunded. However, this is the only thing they can do. SugarCRM does not cover consequential damages, to the extent applicable law allows it.

Additionally, *maintenance* is provided through updates and patches [Suga]. There are no references to *managed release cycles*.

[1] Named SugarCRM in the following

Permissions: SugarCRM offers its subscribers a *perpetual license* [Suga]. *Rebranding* is explicitly forbidden [Sugc].

6.1.2. Intellectual property related features

According to [Suge], several features related to intellectual property can be identified. For a software producer, these features are of importance when it comes to the creation of commercial value.

Software improvements: Sugar Ultimate offers several *functional differences* compared to Sugar Community Edition. For example, it provides "sales forecasting", "marketing reports" and "advanced charts". These features are implemented in an *advanced core product*. In addition to that, *utilities and plugins* are offered as well. These include plug-ins for IBM Lotus Notes and Microsoft Outlook. Furthermore, SugarCRM offers applications for smart phones which allow remote control of the software ("Sugar Mobile", "Sugar Mobile Plus").

Sugar Ultimate shows *improved behaviour*, thus providing *nonfunctional differences* as well. For example, "productivity" is increased through "multi-tasking" while "advanced password management" leads to better "security". Although the software support databases such as Microsoft SQL Server, Oracle and MySQL, no explicit statements on *certification* can be found.

Software distribution Since SugarCRM is a software producer and offers its own software, *software distribution* in terms of Chapter 5 is not provided.

Documentation SugarCRM publishes *documentation* for all four commercial offerings, the free version, the mobile programmes and its plug-ins [Sugf]. These documents are available online and can be accessed without any costs, therefore not constituting a commercial feature. In terms of documentation types, it can be distinguished between "application guide" and "developer guide" [Sugh].

6.1.3. Service features

In addition to its software, SugarCRM also offers services around it.

Support: SugarCRM provides a subscription-based support [Sugb]. There are several channels which can be used to contact support.

First of all, support can be consulted online via the homepage of the company. Paying customers can do so by submitting a case in the "customer self-service portal", consulting articles in the knowledge base, adding an entry to the bug tracker or posting on the forums where help is provided by both employees of SugarCRM and volunteers [Suge]. However, the

last two choices are also available to users of the free version. Except for the knowledge base, all other options belong to the category of *managed support* as they involve interaction with other people. Secondly, users can get support via email [Sugm]. This option is available to everybody – although requests of paying customers are prioritized [Sugb]. Finally, users of Sugar Ultimate can request immediate help by using the phone.

In terms of quality, Sugar Ultimate provides an availability of 24/7 and a response time of one hour. Although the number of incidents is unlimited, quantitative restrictions apply since the number of authorized contacts is limited to five [Sugb].

Training: SugarCRM provides self-study training through videos and tutorials on its homepage [Sugn]. However, some of these videos are available for free.

Paid training is provided by "authorized learning partners" of SugarCRM rather than by the company itself. These training courses are led by an instructor and can either be online (webinar) of offline in a classroom. In addition to that, learning materials are provided [Sugi] [Sugj].

No *professional certification training* is offered.

Client-specific services: Neither *implementation* nor *certification* are offered.

General services: SugarCRM provides paid *software operation* services including *installation*, *configuration*, *data migration* and *hosting* [Sugd]. In terms of *hosting*, SugarCRM offers "Software-as-a-Service" with a specific amount of storage capacity [Suge]. Alternatively, customers may buy this feature from one of the partners of SugarCRM.

In addition to these features, *consulting* is offered through its consultant partners [Sugk].

6.2. OSSCube LLC (Service provider)

OSSCube LLC[2] defines itself as "a leading global Open Source Software focused company, providing an integrated value chain of services encompassing outsourced software development, product customization and implementation, consulting and training services, on open source products and technologies" [OSSa]. The company offers its services for open source software created by third parties, for example SugarCRM, MySQL, Drupal and Moodle [OSSk]. Additionally, OSSCube provides custom development of applications for Android devices, the internet or social media software, for example [OSSe]. Furthermore, the company offers custom solutions problems related to e-commerce and health care. By the definition given in this thesis, OSSCube can be considered an open source service provider.

[2]Named OSSCube in the following

6.2.1. Legal features

As OSSCube does not own the intellectual property rights to SugarCRM and other products it capitalizes on, it can provide neither *commercial licenses* nor special *permissions* for them. On the contrary, it might do so for the results of its custom development services. As stated in [OSSe], *maintenance* is offered for their "enterprise portal solutions". However, further details on licensing could not be found on the homepage.

6.2.2. Intellectual property related features

By creating addons and plugins for third party open source software such as MySQL and SugarCRM, OSSCube also creates intellectual property [OSSj]. Although these software parts are the result of a *client-specific service* ("product customization and implementation" [OSSa]), they are listed here since OSSCube is likely to deploy one solution multiple times (Any implementation which increases the performance of client A's MySQL installation will achieve the same for the MySQL installation of client B).

Software improvements For example, OSSCube offers custom implementation for SugarCRM in order to add "business specific functionalities and integration with legacy / ERP / accounting applications" [OSSm], thus realizing *functional differences*. Since these changes apply to the original software, the results of the implementation process are part of the category *advanced core product*. Furthermore, OSSCube provides custom implementation aiming at analysis and elimination of performance problems in software like MySQL [OSSi]. Consequently, the resulting software changes qualify as *improved behaviour* and therefore as *non-functional differences*.

However, these improvements are not part of a ready-to-be-sold software product. Furthermore, they take place individually and "on-demand" through the deployment of on-site specialists [OSSi].

Software distribution OSSCube does not offer software distributions – however, it offers integration services.

Documentation Although not explicitly mentioned on the homepage, it is likely that OSSCube provides documentation for its developed products since it also offers maintenance.

6.2.3. Service features

Being an open source software provider, OSSCube offers several services in order to generate revenue.

Support: There are two possible scenarios for support offerings. First, OSSCube can provide paid support for its custom implemented solutions. For example, it offers 24/7 support for its "enterprise portal" [OSSe].

Secondly, the company may also provide support for the open source products of its partners. As an example, OSSCube offers support for Acquia's Drupal software. In this case, three different support options are available: "basic", "professional" and "enterprise" [OSSb]. All three options can be considered *managed support* as requests are handled either via "moderated subscriber forums", a "web-based case management" system, email or phone. However, not all channels are available for each option. In terms of quality, initial response times range from eight hours (professional) to two hours (enterprise). No details on quantitative limitations are provided.

Training: OSSCube offers instructor-led on-site training for several products and technologies such as MySQL, Moodle, Ruby on Rails, Apache Hadoop and PHP [OSSf]. In regard to PHP, *professional certification training* is also offered [OSSl].

Client-specific services: As previously mentioned, OSSCube provides *custom implementation*. Examples for this service include the development of Android applications, Linux programmes, applications for social media such as Facebook and complete e-commerce solutions [OSSd]. No details on *client-specific certification* could be found.

General services: Similar to support, OSSCube offers *consulting* for both custom solutions and software of partner companies. Exemplary products include the results of the "enterprise portal development" service and SugarCRM [OSSe] [OSSm].

Features from the category of *software operation* are also available. For example, *installation* ("implementation") and *integration* of software like SugarCRM and Moodle is offered [OSSg] [OSSn]. Additionally, customers may request *(data) migration* from previous IT to MySQL, Drupal or SugarCRM [OSSh] [OSSc] [OSSm]. Furthermore, OSSCube also offers hosting for its custom e-commerce applications and other software like Moodle [OSSd] [OSSg].

6.3. Red Hat, Inc. (Distributor)

Red Hat, Inc.[3] offers a product called Red Hat Enterprise Linux (RHEL), which is a ready-to-use Linux distribution made from a Linux kernel and several other software components [Redp]. Although Red Hat does not hold the intellectual property rights for most of these components, it does so for the configured distribution, thus being able to charge for it [Reds]. The source code of the distribution as well as the code of the underlying Fedora distribution can be accessed for free [Rede] [Redq]. Consequently, Red Hat complies with the definition of an

[3]Named Red Hat in the following

open source distributor. However, Red Hat can also be considered a software producer since it offers a software called "JBoss Enterprise Middleware" [Redh]. Nevertheless, the focus of this thesis will be on Red Hats role as a distributor. Besides from offering the distribution itself, Red Hat also provides services around Red Hat Enterprise Linux with a subscription-based payment model [Redy].

6.3.1. Legal features

Commercial offerings of Red Hat include legal aspects which do not apply when using the distribution for free.

Commercial license: Although Red Hat Enterprise Linux itself is licensed under the GPL, a commercial agreement offers further rights and warranties to paying customers. However, the use of the GPL does not allow *non copyleft usage rights* for the distribution. Instead, such rights can be purchased for another product, Cygwin [Redd]. In terms of *indemnification* and *warranty*, Red Hat addresses this issue explicitly by guaranteeing that "the Services will be performed in a professional and workmanlike manner by qualified personnel" and that "Red Hat branded Software does not, at the time of delivery to Client, include malicious or hidden mechanisms or code for the purpose of damaging or corrupting the Software" (both from [Redo]). Similar to SugarCRM, however, Red Hat does not provide any additional warranties and tries to limit its liabilities as far as possible. If warranty is violated, Red Hat is obligated to resolve the problem, e.g. by conducting maintenance. However, if this fails, customers may cancel their subscription and reclaim pre-paid fees.

Furthermore, only direct damages have to be fixed by Red Hat, and the financial value is limited to the price of twelve subscription rates.

In addition to this warranty clause, Red Hat offers a so-called "Red Hat Open Source Assurance" which is a intellectual property warranty [Redu]. Red Hat guarantees that software is replaced if intellectual property rights of third parties are infringed.

Additionally, Red Hat offers *managed release cycles* by providing stable releases with an interval of "approx 18 months" [Rede] between two consecutive releases. Subscriptions also include *maintenance* and *updates* [Rede].

Permissions: No details on *permissions* could be found.

6.3.2. Intellectual property related features

This section addresses intellectual property which is generated through the configured distribution and the actions which were applied to its individual components. Since the distribution itself is available in the form of source code, users can benefit from this added value even without paying money [Redq].

Software improvements Compared to Fedora, Red Hat Enterprise Linux offers several *non-functional differences*. For example, Red Hat provides *hardware certification* by testing specific hardware for the use with the distribution [Rede]. Systems which were successfully tested are included in the Red Hat Hardware Catalog, thus allowing customers to find information on compatible hardware components [Redv]. Since Red Hat sells an operating system, it does not certify its product against other software. Instead, companies like Acronis, Inc. and Zend Technologies, Inc. certify their products against the Red Hat distribution, therefore creating an additional benefit for paying customers of Red Hat [Redk].

Furthermore, customers may buy "add-on functionality" [Redj] which increases scalability, availability and performance, thus being an example for *improved behaviour* [Reda]. In addition to that, features like the "Smart Management Add-On" improve operational comfort.

Software distribution Red Hat Enterprise Linux is a prime example for a *software distribution* since it integrates a Linux kernel and several other components into a configured software product. These components include a graphical desktop environment, Gnome, a window manager, X.org and applications such as Firefox and OpenOffice [Reds]. They are not owned by Red Hat and are distributed under open source licenses. Consequently, using Red Hat Enterprise Linux is beneficial to customers since it saves them a lot of configuration and integration effort.

Documentation Red Hat provides several types of *documentation* on its homepage, including a "deployment guide", a "developer guide" and "technical notes" [Redr]. However, these resources are not part of a commercial offering as they are available for free.

6.3.3. Service features

Service offerings are a major incentive for spending money on the distribution since the software itself can be obtained for free from Red Hat's ftp server [Redq].

Support: Red Hat provides its subscribers with two different support offers: "production support" and "development support". Additionally, optional support packages such as "technical account management" and support for certified hardware are also available [Redf].

Developer support comes in two options, "professional" and "enterprise". It allows requests via phone and online via the "Red Hat Customer Portal" [Redf]. Consequently, it can be considered *managed support*. In terms of support quality, availability is guaranteed during "standard business hours" and response times range from two business days to four business hours (depending on the option). Quantitative restrictions apply as well. For example, the number of authorized contacts is limited to one. However, the number of incidents is unlimited.

Production support allows the choice between two commercial options, "standard" and "premium" [Redi]. Both are *managed support* and allow requests via phone or online. When looking at support quality, both are available during "standard business hours". However, premium production support offers 24/7 availability for urgent issues, i.e. requests with severity level one or two. Response times depend on the severity level of the request, whether it is standard or premium support and whether it is an initial or follow-up response. The best response time is granted for premium support and severity level one, with both initial and follow-up response time being one hour. No quantitative restriction apply, since the number of incidents is unlimited.

Technical account management represent a *managed* on-site support offering [Redw]. By purchasing this support option, customers receive help from a technical expert which visits them on-site.

Additionally, Red Hat provides support for hardware which is certified against its product [Redg]. Red Hat also offers support through its "Support Essentials" portal where customers may post issues. However, it can be read from non-paying users as well.

Training: Red Hat sells *training* for its software distribution. For example, instructor-led training is offered, either in a classroom ("Classroom Training"), in the customer's company ("Onsite Training") or online ("Virtual Training", "Remote Classroom Training") [Redx]. Additionally, online self-study training is provided through an e-learning application ("Self-paced Training").

Furthermore, *professional certification training* is offered. Customers may learn how to become a "Red Hat Certified System Administrator" or a "Red Hat Certified Architect", for example [Redx]. In addition to that, Red Hat also offers specific certification training aiming at security issues [Redb].

Client-specific services: "Custom engineering services" describe a wide range of *client-specific services*. In addition to *custom implementation*, Red Hat provides services for all phases of the software life cycle, from analysis and specification to deployment, feedback and support [Redc].

No data concerning *custom certification* could be found.

General services: Red Hat provides comprehensive *consulting* services for its software. For example, the company analyses the individual situation of its customers, demonstrates possible usage scenarios for its software and offers advice on efficient deployment [Redm]. Furthermore, Red Hat shows how customers can improve specific areas of their business by presenting the advantages of "cloud computing", "virtualization" and "service-oriented architecture (SOA)", for example [Redn]. As a result, customers receive advice on how to create a more efficient IT infrastructure and a productive operating environment.

In regard to *software operation*, customers can pay for *installation*, *configuration* and "optimization" of Red Hat Enterprise Linux [Redm]. Additionally, Red Hat offers "performance analysis", improvement of security and *(data) migration* through its "strategic migration planning" [Redn].

Although Red Hat supports cloud computing through several technologies and products such as "CloudForms" and "OpenShift", it does not provide *hosting* by itself [Redl]. However, hosting can be purchased from authorized hosting partners of Red Hat [Redt].

7. Empirical study

As already mentioned in Chapter 1, a quantitative survey is the second major contribution of this thesis. The goals of the survey and details on its structure, publication and analysis can be found in Chapter 4. Therefore this chapter focuses on presenting the results of the survey.

7.1. Responses and general statistics

Between 15.09.2011 and 25.10.2011, 37 responses were received. However, this number includes several answers with either no or invalid data[1], therefore reducing the number of suitable responses to the value of 15. Furthermore, a number of 27 products could be identified from the valid responses. From these 15 records, only eight were filled out completely. These eight records described 16 products, which were later used for finding feature bundles. Table 7.1 contains more information about the number of responses.

Table 7.1.: Number of valid responses and products

	Software producers	Service providers	Distributors	**Total**
Valid responses	11	2	2	**15**
Products	18	3	6	**27**
Products (bundling)	9	1	6	**16**

7.2. Ranking of features by category

One of the goals of this thesis is to rank features based on their importance. This thesis employs two different metrics to achieve this:

First, features can be ranked based on the number of products they are part of. However, since individual features received different numbers of responses, the relative number of occurrences (*frequency*) is used. A frequency of one means that each participant offers the corresponding feature in all of his products. This type of ranking is referred to as *implicit ranking*.

Secondly, the participants of the survey were asked to rank the features for each category. Chapter 4 describes how the answers to these questions were transformed into a value called

[1] Our definition of *invalid* was previously discussed in Chapter 4

relative importance. This value equals one if every participant placed it on the first rank of its category. These rankings are called *explicit ranking*.

For each feature category, the following section compares explicit and implicit ranking to each other. These rankings are illustrated in diagrams which are ordered by the overall importance of features (orange colour). Additionally, they show how individual business models influence the ranking. If the names of business model are followed by numbers, these numbers represent the corresponding sample size. Although the order of features in these two rankings can be compared, it is not possible to compare concrete values of frequency and relative importance to each other. Furthermore, the samples which were used for the rankings have different sizes – depending on the business model and the feature category. Consequently, the order of features may look different in reality. Table D.2 and Table D.4 in the appendix provide an overview of the individual sample sizes.

7.2.1. Legal features

As seen in Figure 7.1 and Figure 7.2, both rankings have *maintenance* and *updates* on the first two ranks. These two features are mainly offered by software producers and service providers. On the contrary, *non copyleft usage rights* can be found at the end of the rankings. Furthermore, it is noticeable that distributors favour *perpetual licensing* and *managed release cycles* in both rankings. However, the results for service providers and distributors must not be overrated as the corresponding sample sizes range from two to six.

7.2.2. Features related to IP

Both rankings in Figure 7.3 and Figure 7.4 regard *additional functionality* and *digital documentation* as the two most important features. Due to the limited sample size in terms of service providers, it is impossible to derive significant conclusions for this business model. The same applies for service providers in the explicit ranking. However, the implicit ranking shows that all three products from distributors are nearly identical. Furthermore, it is not surprising that *software distribution* is important for distributors. The fact that *certification for the use in processes* as well as *certification of development processes* are not considered important by participants is backed up by the observation that not a single product contains these two features.

7.2.3. Support

This section refers to Figure 7.5 and Figure 7.6. Although the majority of participants considers *managed support* the important feature, *unguided support* is the one which is part of every single product. Similar to IP related features, nothing can be said about service providers. In regard to distributors, one has to notice that all three products have the same features.

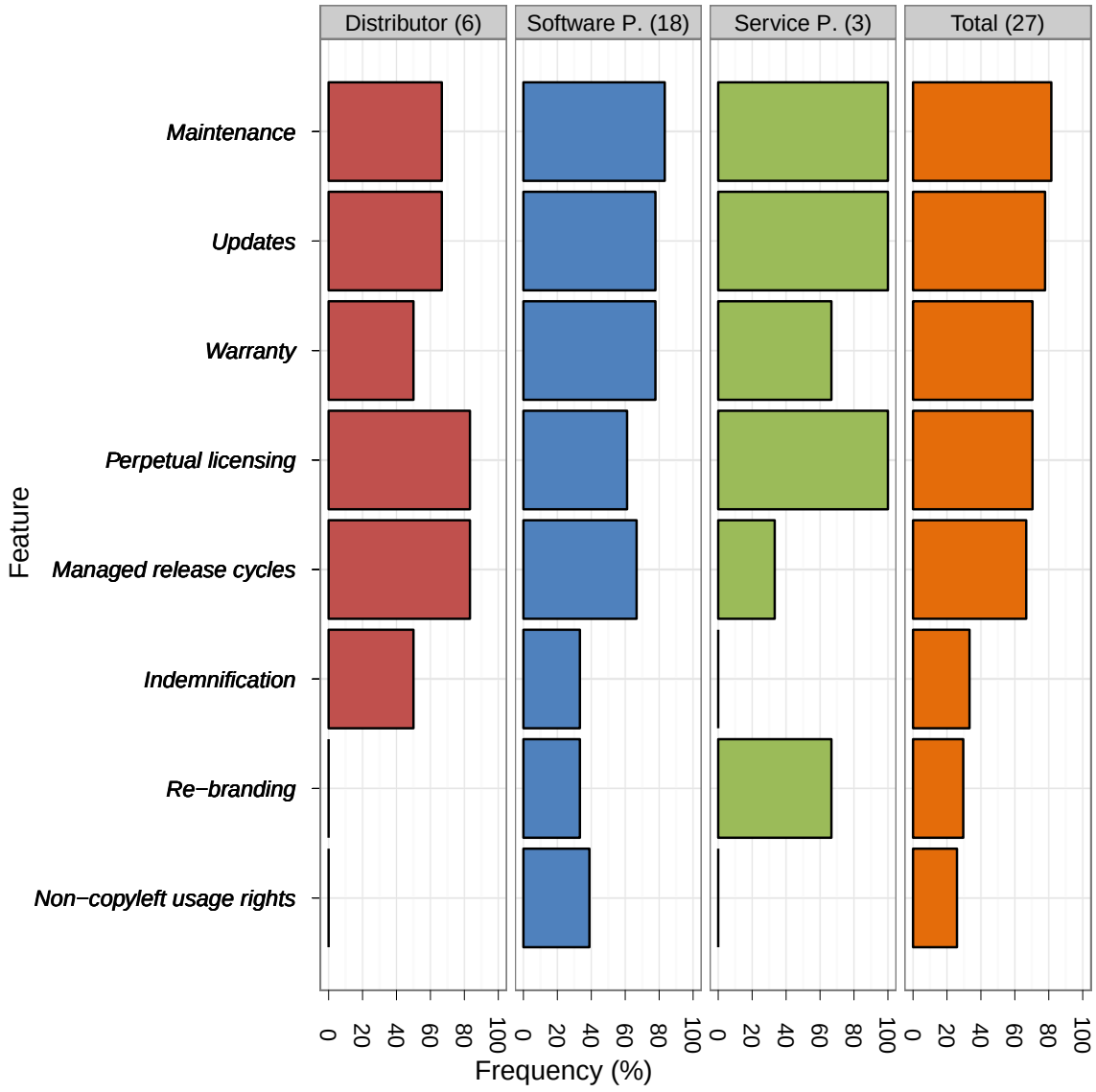


Figure 7.1.: Ranking of legal features based on their frequency

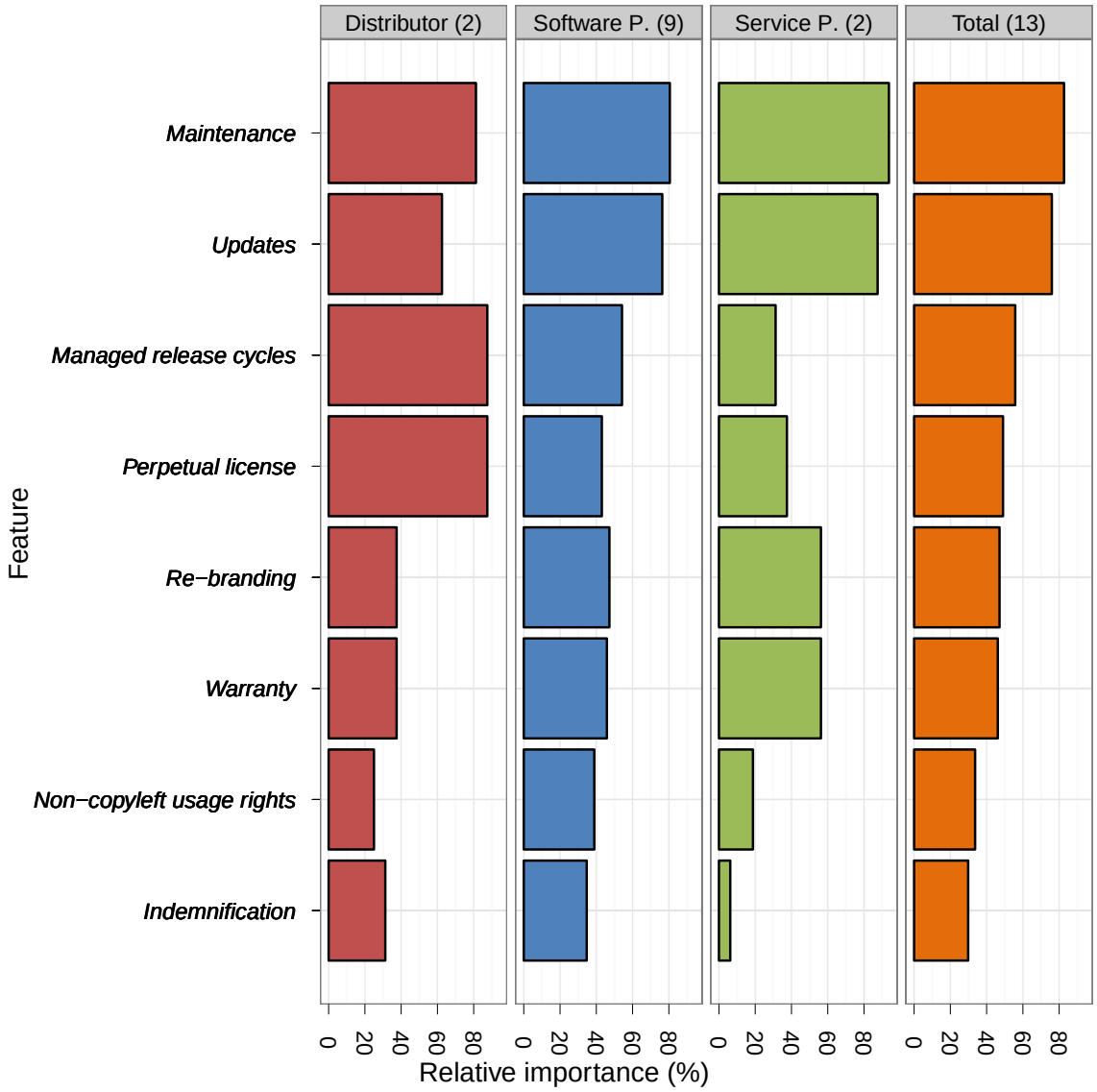


Figure 7.2.: Ranking of legal features based on the participants' opinions

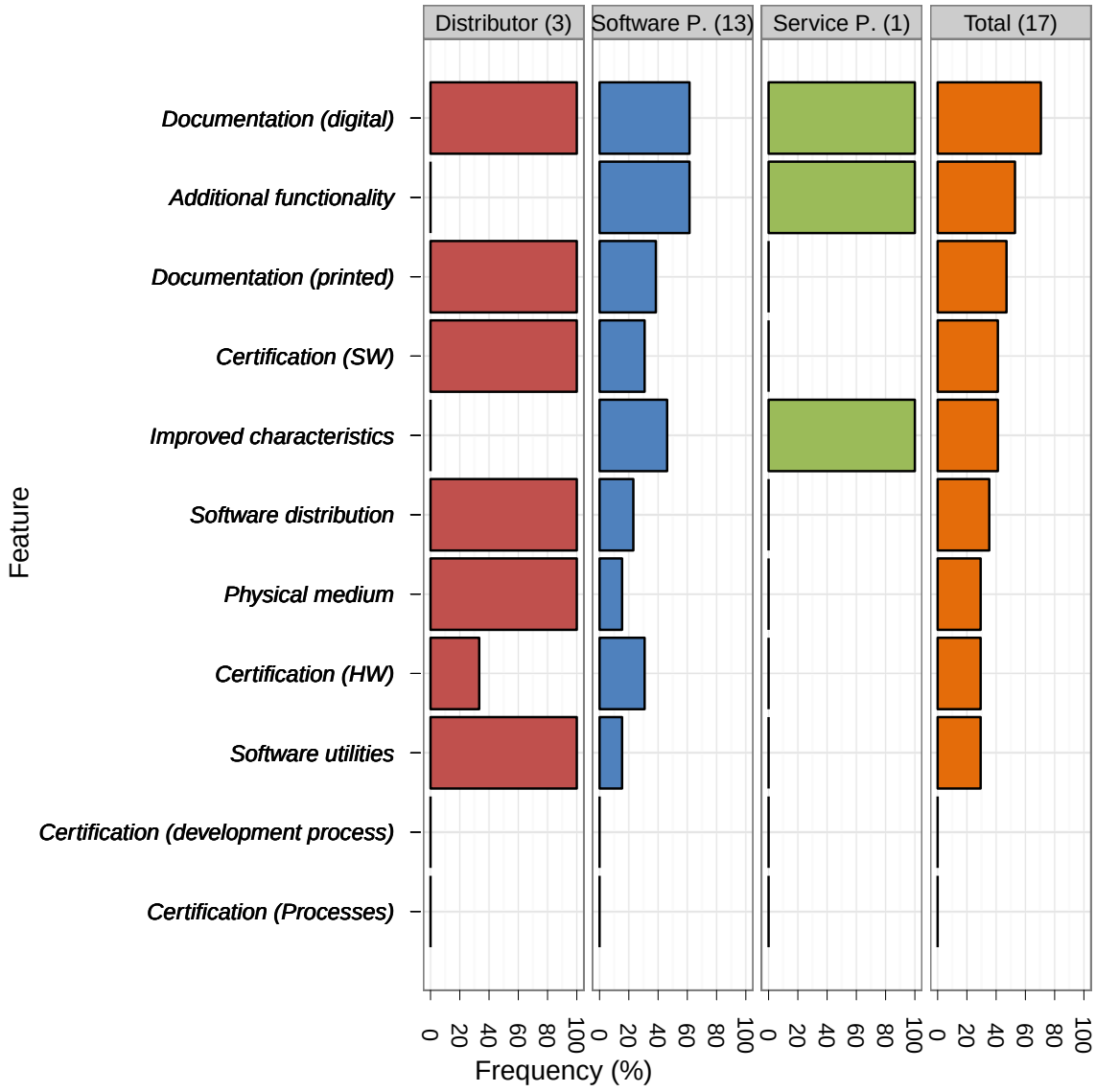


Figure 7.3.: Ranking of IP related features based on their frequency

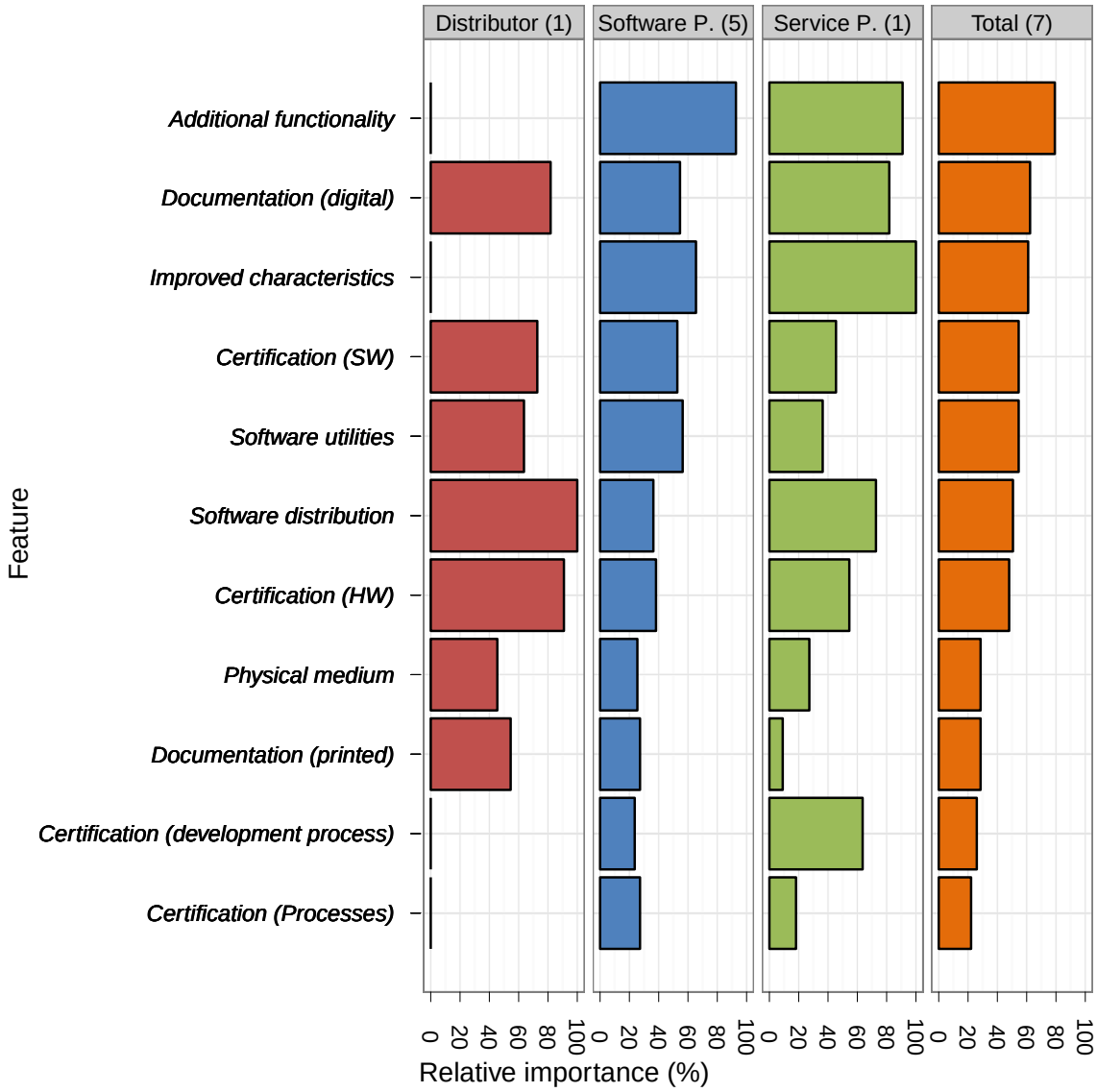


Figure 7.4.: Ranking of IP related features based on the participants' opinions

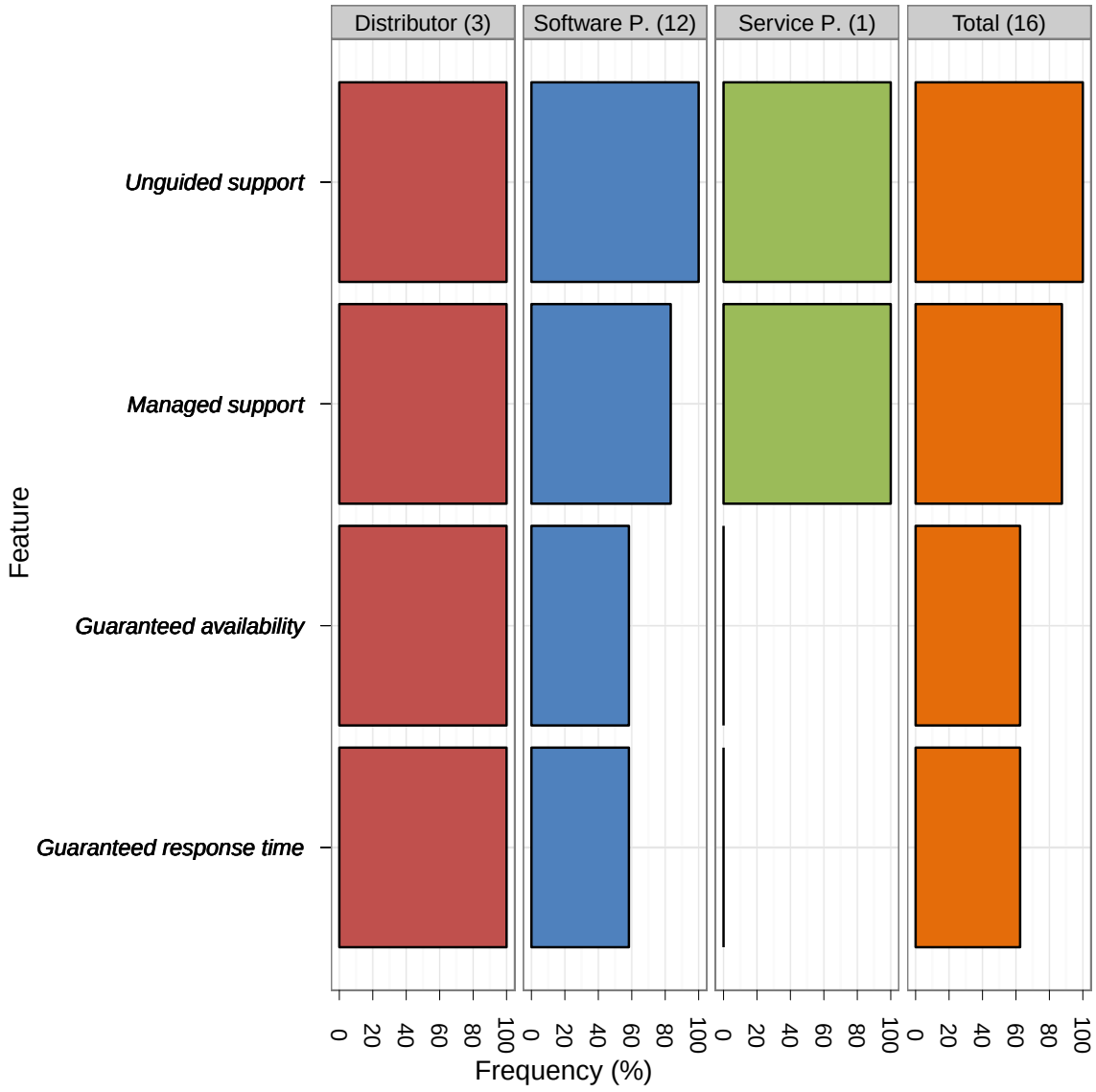


Figure 7.5.: Ranking of support features based on their frequency

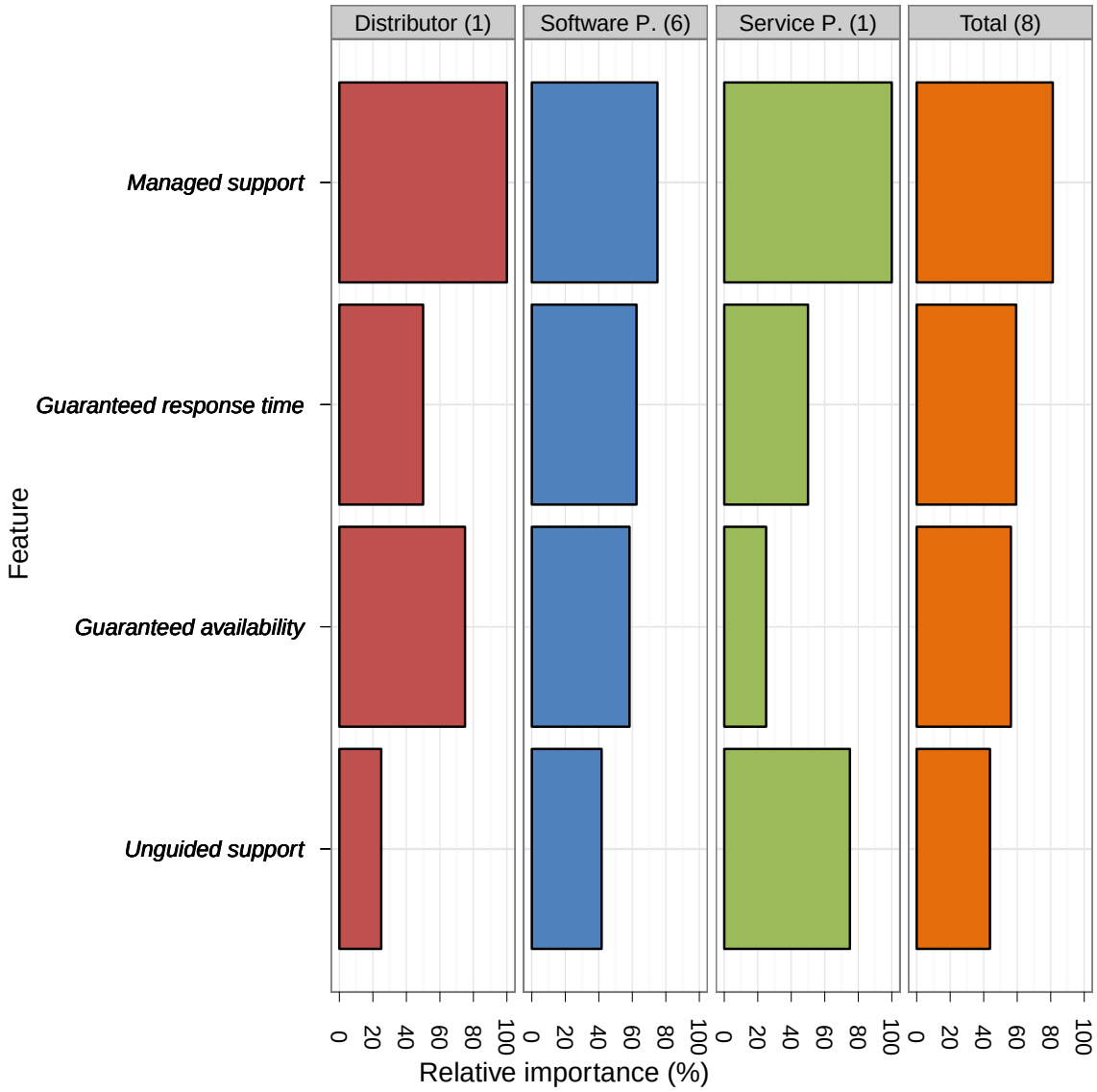


Figure 7.6.: Ranking of support features based on the participants' opinions

7.2.4. Training

As seen in Figure 7.7 and Figure 7.8, both rankings place *on-site training* on the first rank. However, the other three ranks differ from each other. Furthermore, all three products from distributors do not contain *online training*.

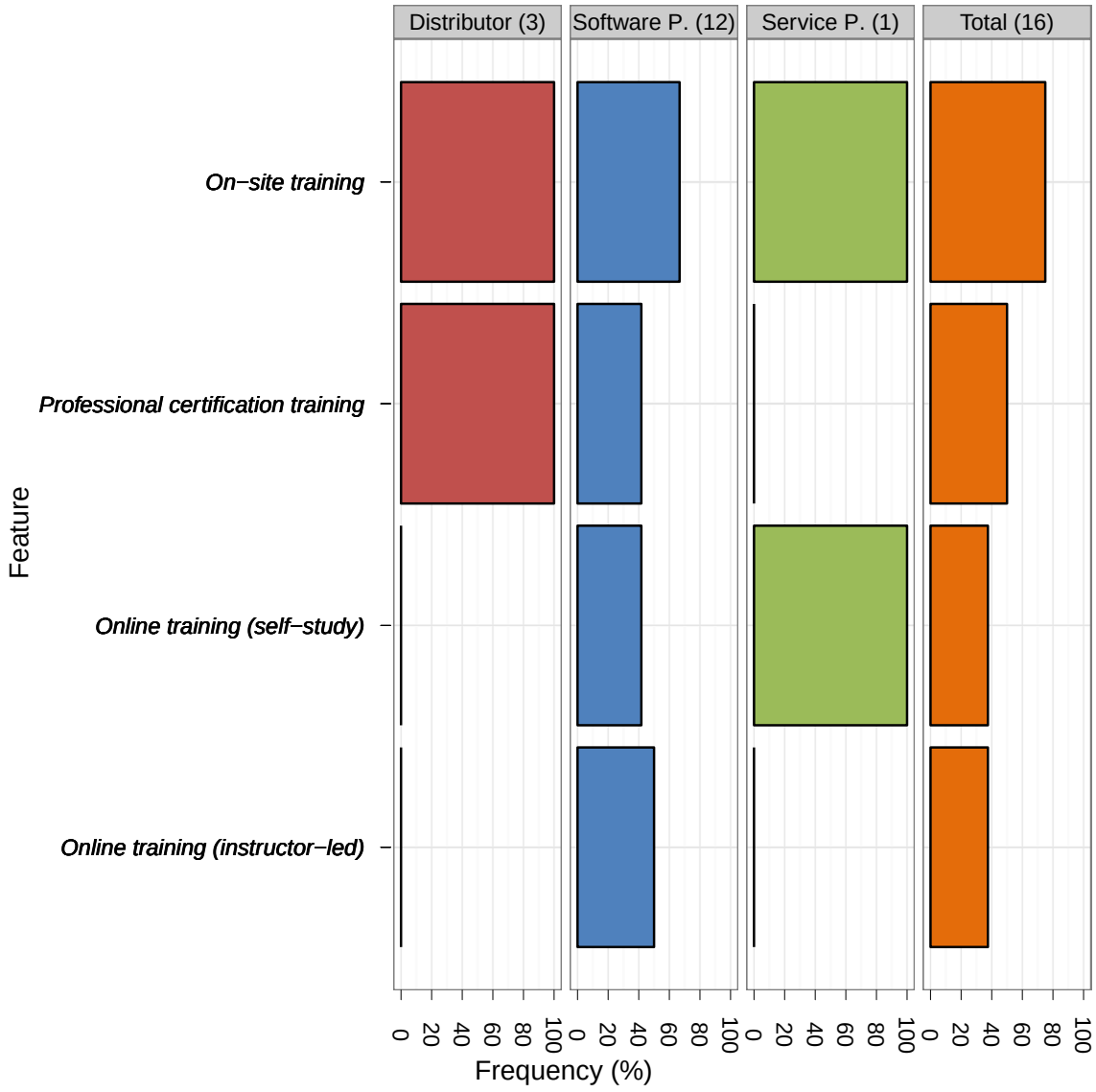


Figure 7.7.: Ranking of training features based on their frequency

7.2.5. Client-specific services

The claim that *custom implementation* is more important than *client-requested certification* is supported by both Figure 7.9 and Figure 7.10. Moreover, it is eye-catching that no client-

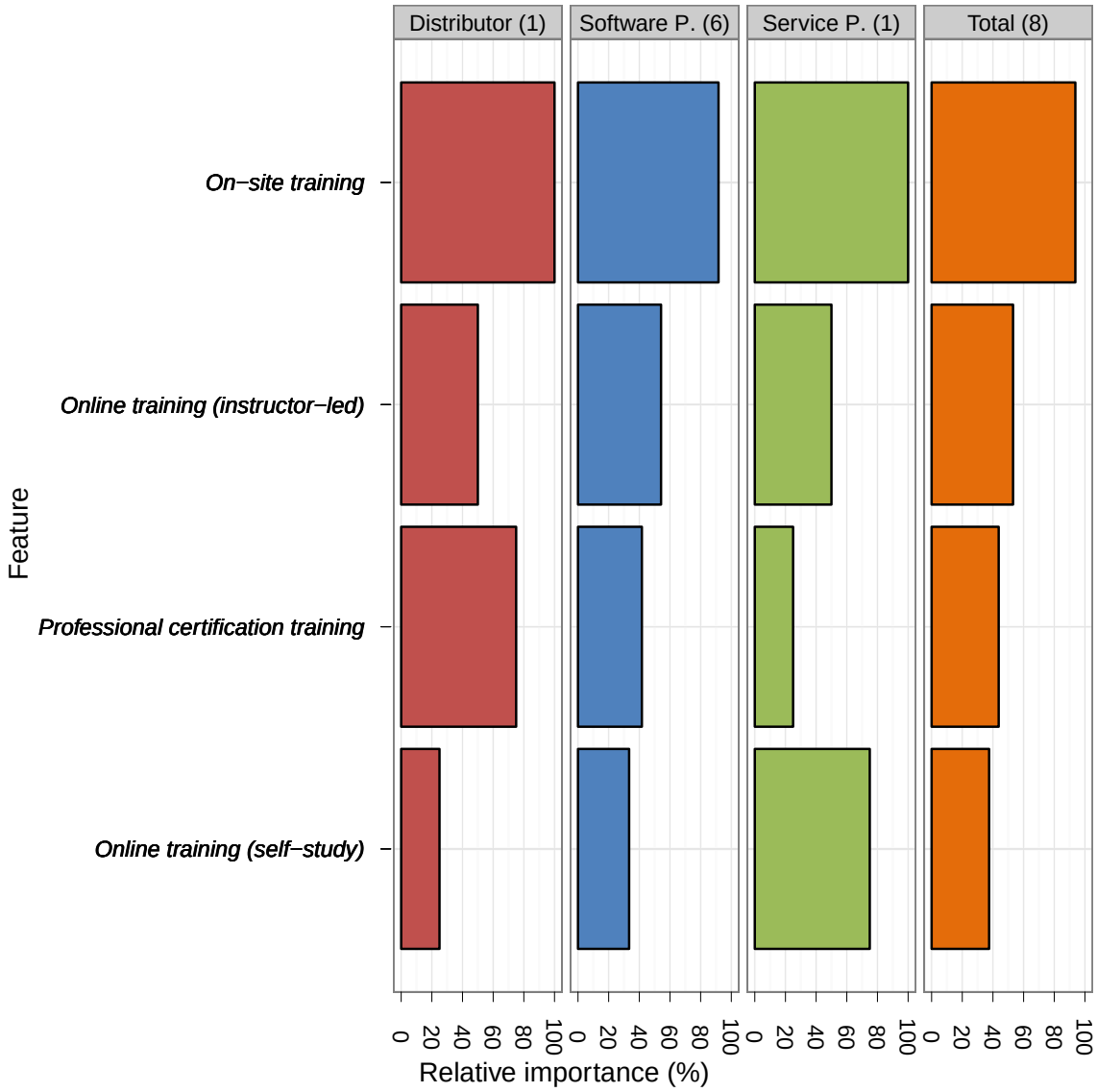


Figure 7.8.: Ranking of training features based on the participants' opinions

specific services are offered for any of the distributors' products while nearly 80 % of the software producers' products contain at least *custom implementation*.

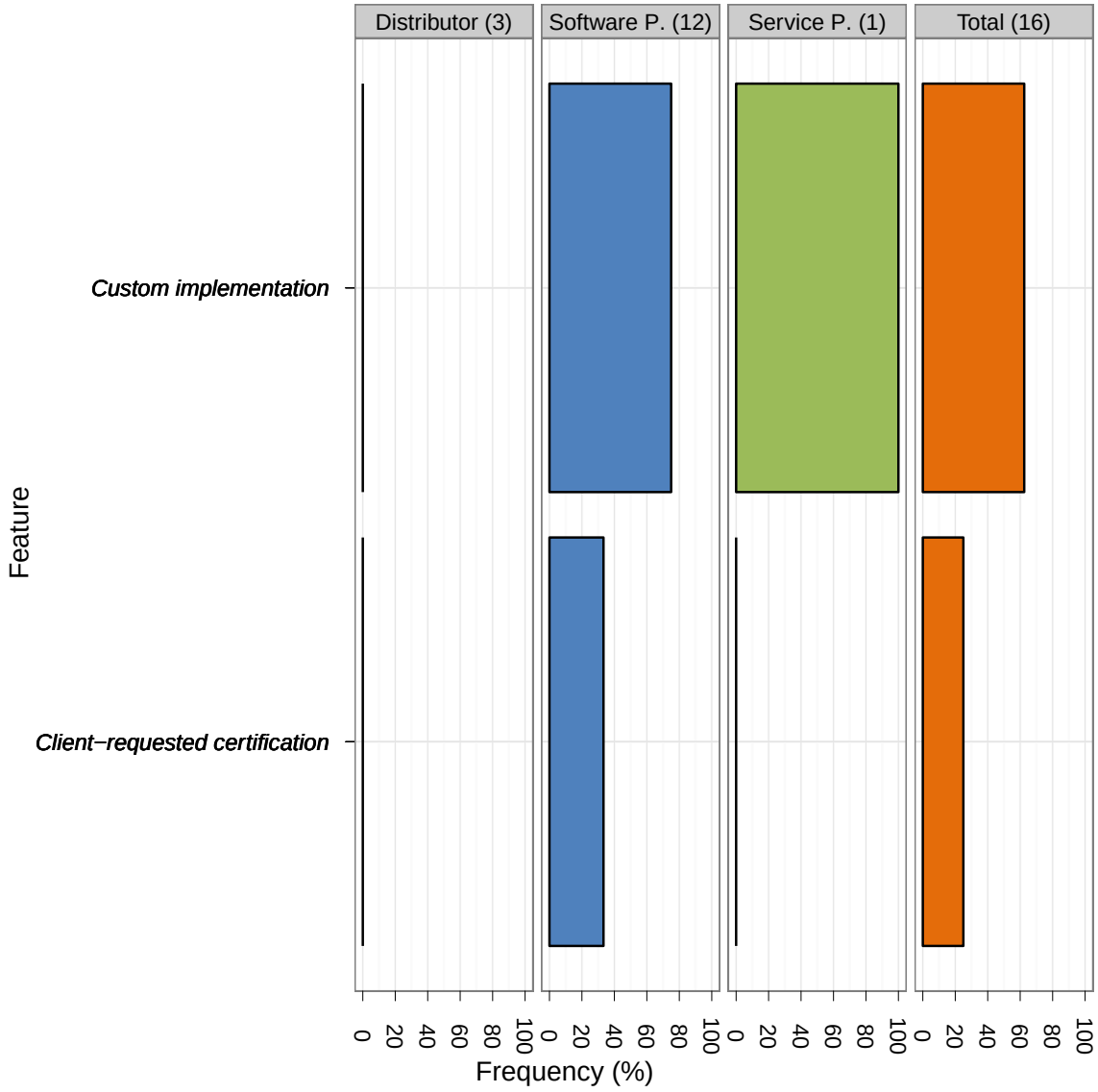


Figure 7.9.: Ranking of client-specific service features based on their frequency

7.2.6. General services

Except for *hosting* and *data migration*, the order of features in Figure 7.11 and Figure 7.12 is identical. Surprisingly, *hosting* is not offered for most products – even though cloud computing is a popular trend. In fact, *installation and configuration* and *installation* seem to be more important. Furthermore, one has to notice that none of the distributors offers general services

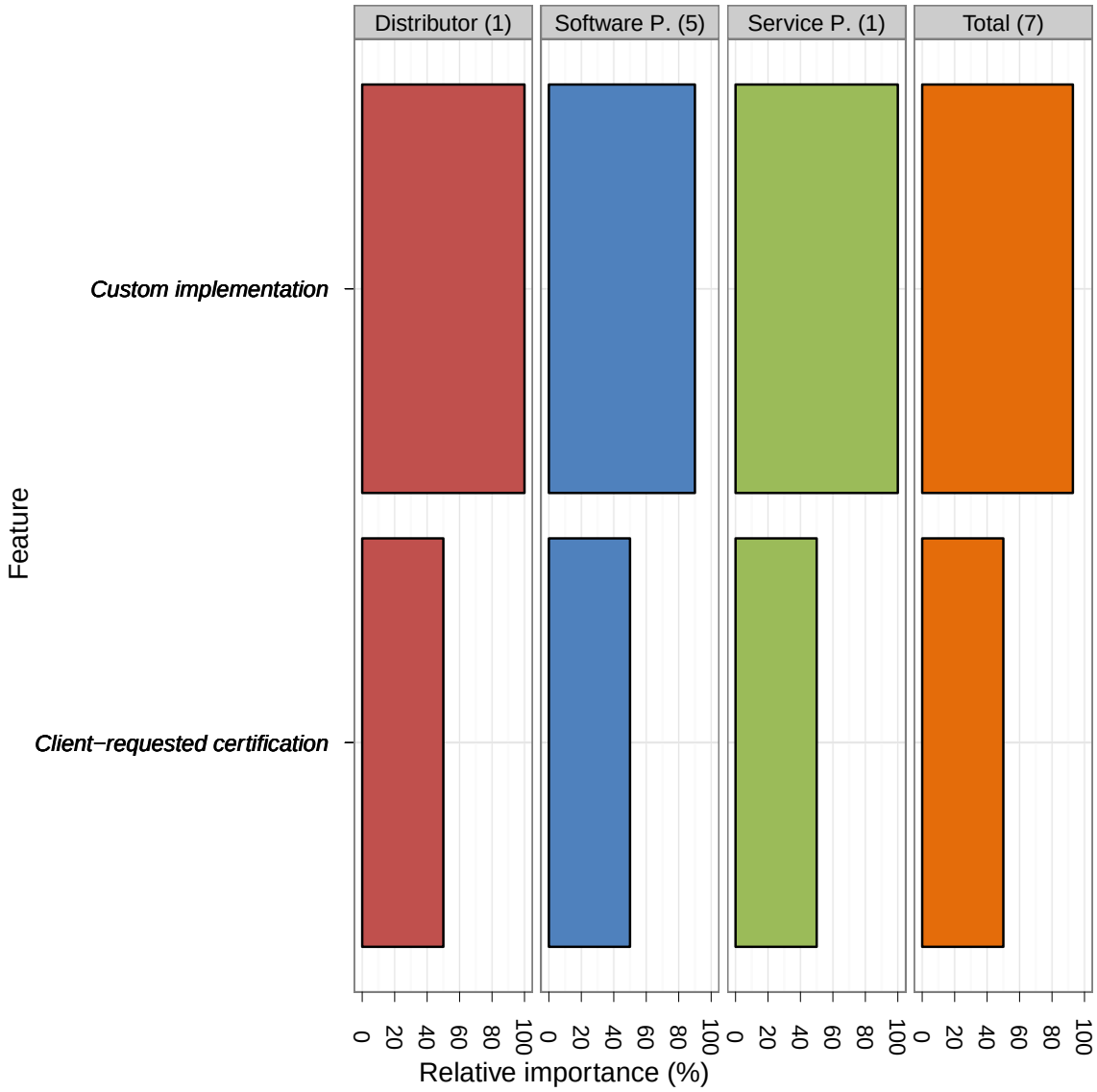


Figure 7.10.: Ranking of client-specific service features based on the participants' opinions

for his products.

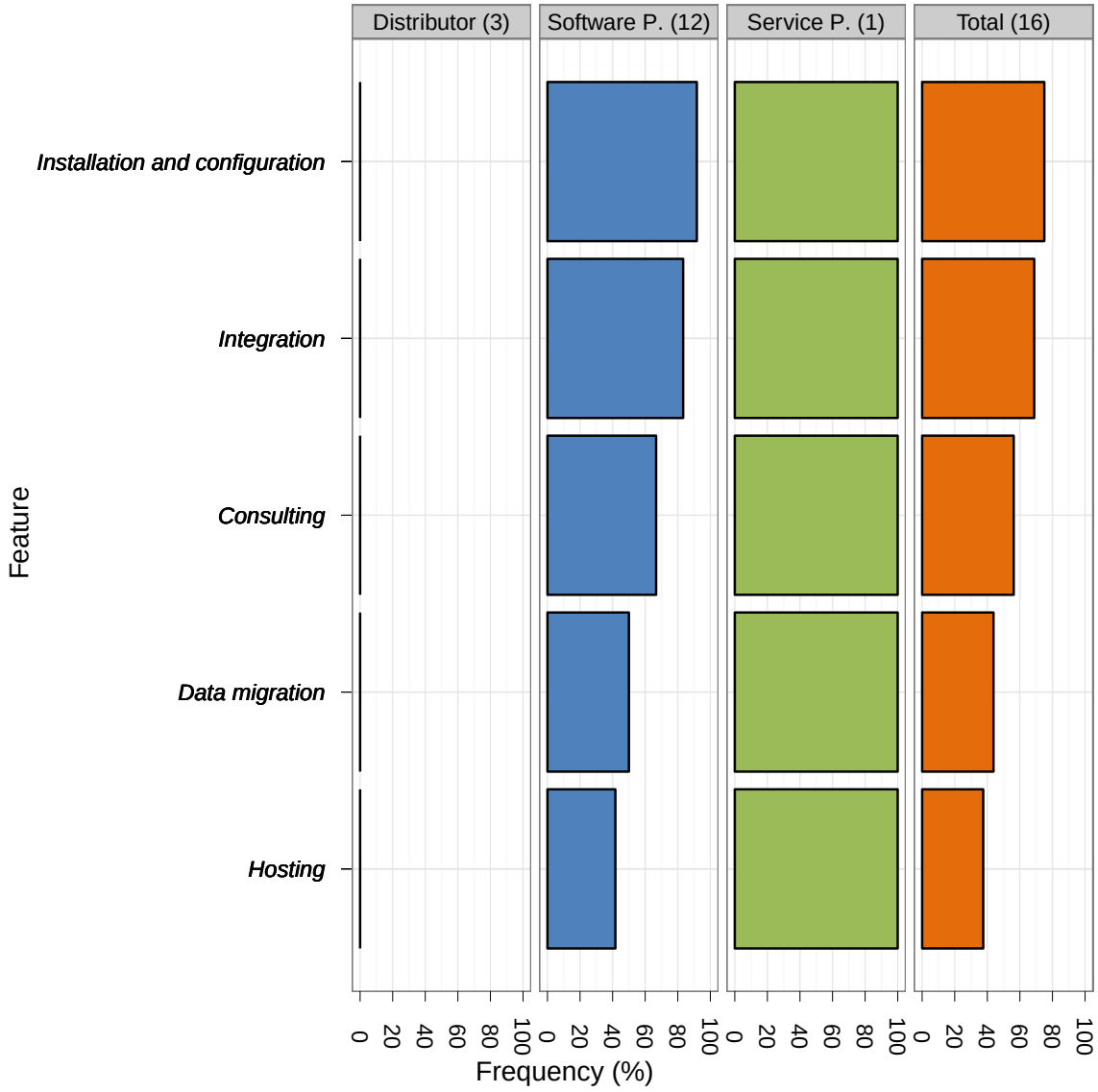


Figure 7.11.: Ranking of general service features based on their frequency

7.3. Frequency of features

In order to compare the frequency of features from different categories, the individual implicit rankings are combined into a single one. Figure 7.13 shows all features ranked by their frequency, both in total and for each business model individually. Additionally, Table D.1 in the appendix contains the exact numerical values for these frequencies. However, comparing these values is difficult since the respective sample sizes differ, depending on business model

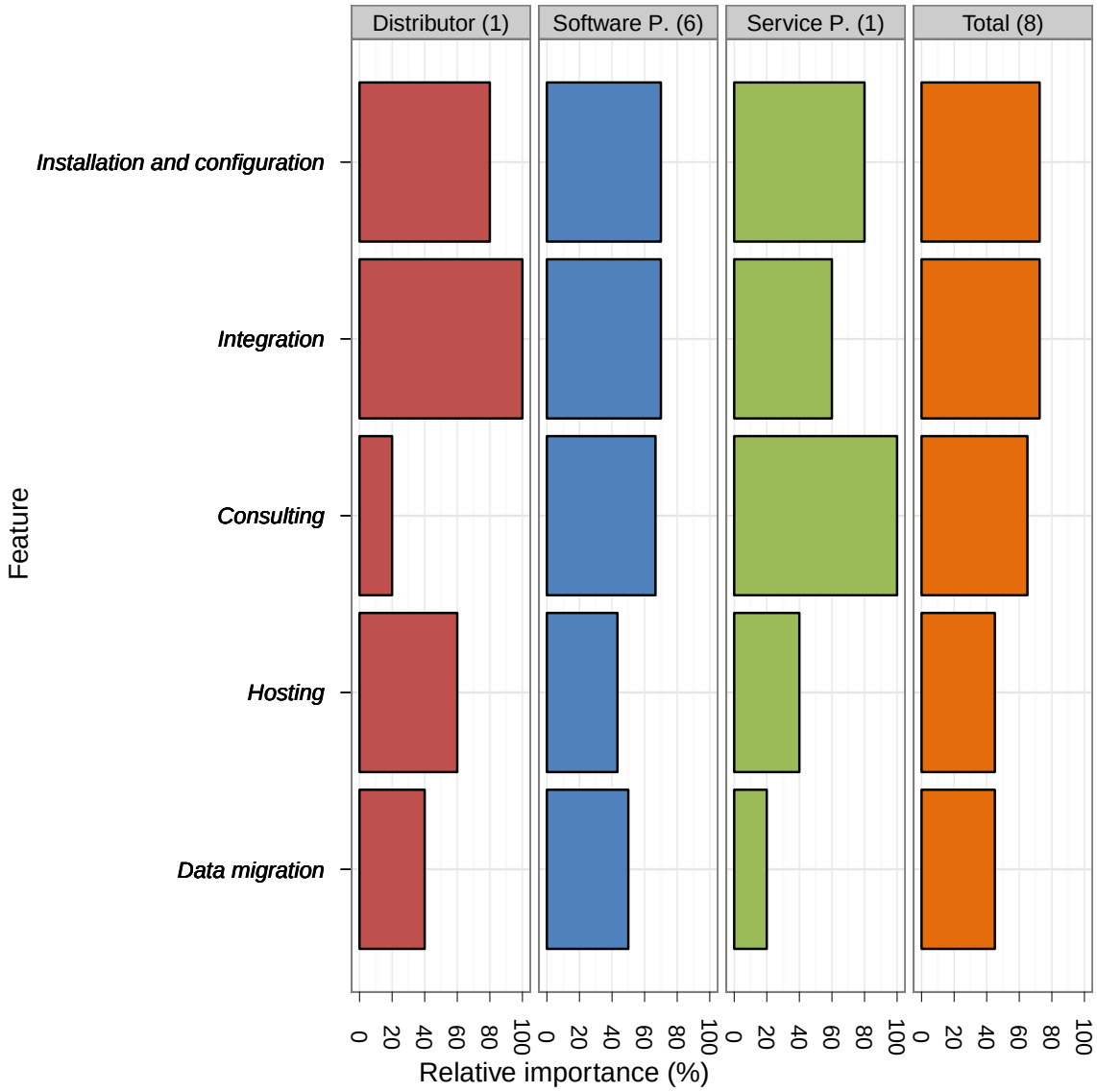


Figure 7.12.: Ranking of general service features based on the participants' opinions

and category. Table D.2 in the appendix gives an overview of these different sample sizes. Furthermore, Table D.3 contains the corresponding confidence intervals (0.95 level).

Judging by the total ranking, most products contain features which ensure that the software keeps on working. *Support* (both *unguided* and *managed*), *maintenance* and *updates* are the most important features. Furthermore, it is noticeable that the physical availability of software and documentation is not very relevant. Moreover, one can see that neither *certification for the use in processes* nor *certification of development processes* are offered at all. In terms of *non-copyleft usage rights* and *client-requested certification*, these features are only part of products from software producers. This observation is surprising – after all, *non-copyleft usage rights* is the only feature which is exclusive to the domain of open source software. Consequently, we had expected it to appear more often. When looking at service providers and distributors, it is eye-catching that a significant amount of features is included in every product. However, sample sizes for service providers range from one to three while distributors received between three and six responses. Consequently, one has to be careful about statements concerning distributors while nothing can be said about service providers. Due to this fact, it is not possible to find out which features are especially favoured by certain business models.

7.4. Bundling

As discussed in Section 4.5, 16 product descriptions were used to find common patterns of features (*bundling*) within commercial offerings. These products originate from the eight complete responses to the survey.

By iterating over every possible combination of individual features, some bundles can be identified. Since there are more than 17 billion possible bundles, only a selection is presented here. Only one product from a service provider and three products from distributors were recorded, therefore no bundles for individual business models can be found. Moreover, we omitted redundant bundles, i.e. a bundle X consisting of features A and B is only considered if (a) there is no bundle Y with features A, B and C or (b) Y exists, but it is part of less products than X is. Each of the following bundles either appears in most of the products or has a high number of features.

Bundle #1: *Warranty*, *maintenance*, *managed support* and *unguided support*

This bundle is the most important one since it is part of **14** products. There are two possible reasons why this combination appears that often. First, these features are strongly connected to each other. As seen in Chapter 5, *maintenance* is a possible consequence of *warranty*. The other two features illustrate all potential forms of *support*. Secondly, combining these four features is reasonable since all of them ensure that software can be operated without problems.

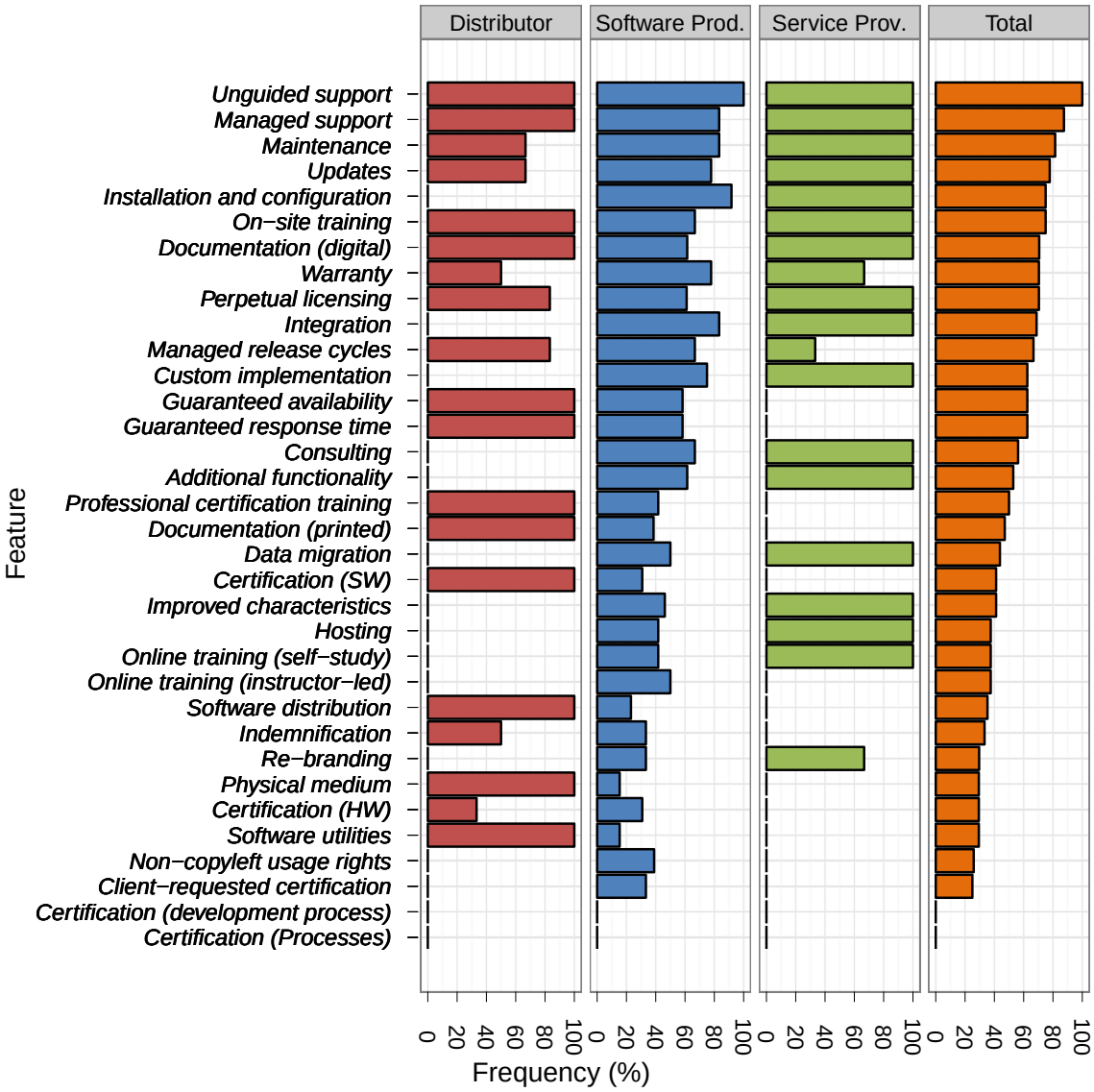


Figure 7.13.: Ranking of all features based on their frequency

Bundle #2: *Updates* and *unguided support*

A total of **13** products contains this bundle. Similar to the first combination, both features allow smooth software operation. Consequently, it is rational to offer such a bundle.

Bundle #3: *Warranty*, *maintenance*, *managed support*, *unguided support* and *on-site training*

This bundle is almost identical to the first one. However, it has one more feature (*on-site training*), but appears only **12** times. Its existence can be justified in a similar way: Since *on-site training* empowers employees to deal with problems, they can help to ensure smooth software operation as well.

Bundle #4: *Managed release cycles*, *digital documentation* and *unguided support*

This combination is part of **12** products. Both *digital documentation* and *unguided support* help during the use of software. *Managed release cycles*, however, do not have a relation to the other features.

Bundle #5: *Unguided support* and *installation & configuration*

12 products contain these two features. While the second feature aims at the initial deployment of software, the first one ensures its smooth operation. It is noticeable that *managed support* is not part of this bundle.

Bundle #6: *Perpetual licensing*, *warranty*, *maintenance*, *managed support*, *unguided support* and *on-site training*

This bundle can be found in **11** products. It contains both legal and service features. As mentioned before, all of them aim at the "smooth operation" of software – except for *perpetual licensing*. The latter one indicates that this bundle is subscription-based.

Bundle #7: *Warranty*, *managed release cycles*, *maintenance*, *digital documentation*, *managed support unguided support* and *on-site training*

Although this bundle is part of only **10** products, it is listed here since it contains seven features. No bundles with eight or more features can be derived from the responses. It is eye-catching that its features follow a similar trend as indicated by previous bundles.

Bundle #8: *Warranty*, *maintenance*, *updates*, *managed support*, *unguided support*, *guaranteed response time* and *guaranteed availability*

Similar to the previous bundle, this one is mentioned here because it consists of seven features and is part of **10** products. All these features aim at the smooth operation of software. Furthermore, relations between these features exist. According to the model, *warranty*, *maintenance* and *updates* can be grouped together since warranty may imply maintenance, while updates are a possible part of maintenance. The other four features refer to both types of support and its qualitative characteristics. Once again, this bundle indicates that there is a trend towards the bundling of "smooth operation" features while others like *additional functionality* are not that popular.

7.5. Empirical evaluation of the model

The evaluation of the model bases on the implicit ranking of all features (see Figure 7.13 and Table D.1). Table 7.2 contains statistics about the quality of this ranking. Both mean and median are around 50 percent, while the standard deviation is roughly 25 percent.

From a total of 34 possible product features, only two of them were not part of a single product. These two are *certification for the use in specific processes* and *certification of the development process*. However, "normal" software – such as customer relationship management software or content management systems – without special security requirements is unlikely to need these two features. Consequently, the fact that they were not selected at all is of minor significance. When looking at the other features, each of them has a frequency of at least 25 percent.

In terms of missing features, the survey contains a question explicitly asking for them. All answers to this question can be found in Section C in the appendix since most of them contain annotations to the questions or more restrictions rather than features. However, four missing features can be identified, with all of them being part of the category *training*:

"LPI Certification": We think that this refers to Linux certification offered by the Linux Professional Institute. However, this is already included in *professional certification training*.

"Train the trainer": Obviously, the people providing software training have to be trained, too. This can be addressed by adding another dimension to the category *training*

"Partner enablement" and "partner certification": Finally, these two features may result in another dimension for *training*, therefore leading to three dimensions: training of users, training of trainers and training of partners.

Consequently, these features can be incorporated into the model by extending the dimensions of *training*. However, this does not have any impact on the other features and categories. As a result, we think that the model represents an accurate and comprehensive collection of commercially viable features of open source software. Although the sample sizes are too small to permit representative statements, these results do not refute the model.

Table 7.2.: Statistics for the frequency of features

Maximum	100 %
Minimum	0 %
Mean	50.39 %
Median	48.55 %
Standard deviation	23.60 %
Sample sizes	between 16 and 27
Features at 0 %	2 (5.88 %)

8. Limitations

In this chapter, limitations to the content of this thesis are discussed in detail. Both theoretical work and empirical study have weak points which arise from the research process and its underlying data.

8.1. Theoretical work

When looking at the theoretical part, several possible limitations appear. First of all, the present thesis considers only three business models, therefore neglecting other possibilities such as hardware companies which use open source only to boost their sales.

In terms of the theoretical base, the data constituting the model was gathered from only seven interviews. As discussed in Section 4.1, all interview partners have similar backgrounds, therefore possibly leading to biased data.

Additionally, the significance of the data used to extend the results derived from these interviews is limited as well. These results were extended through the examination of product portfolios of open source companies and through expert discussions. However, the number of companies was small (8) and their selection was arbitrary. Additionally, the discussions about these results took place with only one expert (Dirk Riehle).

Further limitations arise from the process which led to the model. First of all, one cannot tell whether the answers from the interview were in any way influenced by the interviewer. Secondly, the interview analysis might be subject to coding errors as no formal method (such as “grounded theory”) was used.

8.2. Empirical study

Similar to other quantitative studies, representativeness is a major concern when it comes to the survey. Unfortunately, only 15 valid answers were recorded during a time span of two months (see Section 7.1). Furthermore, only eight responses were complete. Since the categories were always presented in the same order, the ones whose questions were located at the end of the survey had an even lower number of responses. Consequently, the results presented in Chapter 7 are far from being representative.

As stated before, 27 products could be derived from a total of 15 responses. However, the actual amount of unique products might be lower than this number due to the fact that

participation in the survey was anonymous. As a consequence, it is impossible to eliminate the possibility that multiple employees from the same company participated in the survey. Therefore the significance of these results is even lower.

Additionally, it remains unlikely that the small set of participants represents all different kinds of open source companies. When looking at the 27 valid product descriptions, for each answer from a service provider, six answers from software producers and two answers from distributors were recorded (18 products from software producers, six from distributors and three from service providers). It is questionable whether this ratio reflects the distribution of business models in the real world. Moreover, the dominance of software producers leads to another problem. If features are more likely to be offered by a software producer than by a distributor, this particular feature will dominate as well – even if reality looks different.

In addition to representativeness, one has to question objectivity, reliability and validity as well, since no formal tests for these criteria were conducted. Consequently, the results may be subject to statistical errors. For example, one participant spotted a possible error in the survey: "There is some level of ambiguity in some of the lists. e.g. some of our customers would very much want us to host our software while we dont do this out of strategic reasons. another freeform comments field below the selection and ranking tables would be appropriate to remedy this".

In regard to the evaluation of the model, possible problems arise since the descriptions in the survey are not completely identical to the ones in the model. As described in Section 4.3, this is due to the fact that the survey has limited space for explanations and definitions. Furthermore, the complete hierarchical structure with all dimensions and details could not be reflected in the survey since it would have increased its length by far.

The process of data analysis and interpretation can also be challenged. We assumed that the participants can judge how important individual features are to their customers – in practice, this statement has to be questioned. Moreover, we reckoned that the participants ranked these features on a linear scale. Our decision to consider rankings which did not have all features assigned can be questioned as well. However, this did not turn out to be wrong as one participant left the following comment on the survey: "There were some items on the list that were not important to clients at all (say 3 were important, but others not at all). So, maybe, if we could rank those individually instead of ranking them amongst each other." Furthermore, the ranking of features might be different in reality since confidence intervals are quite large (around 25 percent in each direction, see Table D.3 in the appendix).

9. Conclusions

This chapter focuses on a concluding reflection of the contents of this thesis. First of all, the results from both theory and empirical work are evaluated. Secondly, we present possible starting points for future research based on this thesis. The final section consists of a summary which recapitulates its major contributions.

9.1. Evaluation

As illustrated in Chapter 8, the explanatory power of this thesis is limited due to several facts. Nevertheless, we believe that the model presented here is an authentic description of commercial open source features and represents a solid base for future research on this topic. This opinion is backed up by the fact that the theoretical work rests upon interviews and discussions with experts who have up to twenty years of experience in the domain of open source. In addition, the model was revised through three iterations which included the analysis of relevant literature. Furthermore, we constructed it with the intention that it was to be evaluated and extended through a survey.

Moreover, the responses to the survey which were considered valid did not refute the model. Although their small number makes this statement not representative, there is at least no contradiction to the model. Additionally, only two features in the model were not selected as parts of any product (*Certification for the use in specific processes* and *certification of development process*). We think that these two constitute rather uncommon features of widespread software – compared to support or training, for instance. Although three aspects are missing from the model, these can be added without any problems. Therefore we are convinced that these facts not render the model void.

9.2. Future work

The limitations of the empirical study provide a starting point for future work on this topic. First of all, it should be a major goal to improve representativeness by conducting another survey with a broader audience and better response rates. For example, incentives for participation could be offered. Additionally, a more exhaustive pilot test should be conducted in order to get more feedback, hence improving the overall quality. Moreover, the survey should be translated into multiple languages in order to reach an audience whose native language is

not English. Finally, it should be considered to promote the survey to a larger audience by using mailings lists and more newsletters, if available.

Since only a low number of participants completed the current survey, its structure and content are likely to need revision. For example, the questions could be designed to make participation more interesting and less monotonous, e.g. by removing the matrix questions which were only asking for facts rather than for personal knowledge. These question could be replaced with more complex questions on individual product features, asking for motivational factors and causal correlation. Additionally, more explicit questions on bundling and relations between individual features could be included. At last, individual categories could be designed so that they do not have an identical structure. These actions would also lead to a better understanding of product design in practice, since the results would reflect more than just the enquiry of product portfolios.

Furthermore, formal statistical tests should be used to ensure reliability and validity of the questions and reduce the impact of statistical errors. Researchers should enhance the process of evaluation by consulting experts with more experience in the field of empirical studies.

In regards to the model of product features, a qualitative improvement could be achieved by conducting additional interviews with a more diverse selection of interview partners. In addition to that, researchers could consult more experts in order to refine and revise the model once more.

9.3. Summary

First, the present thesis outlined the unique characteristics of open source and their impact on commercial adoption. Three major business models for open source companies were identified: software producers, service providers and distributors. These concepts served as a basis for the theoretical part of this thesis, consisting of a comprehensive model of commercially viable product features in open source software. This model was based on the analysis of qualitative interviews, related literature and discussions with domain experts. It classified product feature in three major categories: legal features, features related to intellectual property and service features. The last category was then further divided into four sub categories: support, training, client-specific services and general services. For each feature, we addressed why companies and individuals would spend money on these features rather than using the basic version of the open source product. Even more, the dimensions of these features were discussed in detail, thus illustrating how they could be implemented in a commercial offering. We emphasized legal aspects since differences to traditional closed source software emerged in this area.

Furthermore, we conducted an empirical study in order to evaluate the claims of the theoretical work. Although the number of valid responses was small, we used the resulting data to get evidence on how important individual features are. Depending on the category of features,

both noticeable differences or similarities could be observed when comparing the participants' ranking of importance to the actual frequency of features. Consequently, no final conclusion could be drawn concerning this issue. However, we observed that features such as support, maintenance and warranty were offered in most products. This trend was also supported when looking at so-called feature bundles – specific patterns of features appearing in several products. Depending on the number of occurrences as well as the number of features, the most important bundles were identified. We found out that all of them aimed at the smooth operation of software by containing features such as support, warranty, maintenance and training.

Moreover, we analysed the responses to the survey in order to evaluate the model. Despite the limited representativeness and minor adjustments to the model, we claimed that the empirical data supported the model in general. Next, we highlighted limitations to the content of this thesis. Although both theoretical work and empirical study are limited in several ways, we came to the final conclusion that the results of this thesis constitute a valid base for future work. Finally, we showed possible starting points for future work, including the conduction of an improved empiric study.

10. Acknowledgements

First of all, I am deeply grateful to Professor Dr. Dirk Riehle for his constant guidance during the research process. Additionally, I would like to thank two of his Ph.D. students, namely Hannes Dohrn for his feedback on the survey and Carsten Kolassa for his help in regard to R and statistics in general. Last but not least, I would also like to thank Marco Bögel, Benedikt Lempetzeder and Bert Riffelmacher for their comments on this thesis.

Appendices

A. Sources used during iterative revision

In order to refine the initial model, we analysed the product portfolios of several open source companies. A list of them is part of Table A.1.

Table A.1.: Companies whose product portfolios were used for iterative revision

Company	URL
Alfresco Software Inc.	`http://www.alfresco.com/products/`
Digium, Inc.	`http://www.digium.com/en/products/software/`
Hyperic (part of VMware, Inc.)	`http://hyperic.com/products`
Jaspersoft Corp.	`http://www.jaspersoft.com/editions`
MySQL AB (part of Oracle Corp.)	`http://mysql.com/products/`
OSSCube LLC	`http://www.osscube.com/services`
Red Hat, Inc.	`http://www.redhat.com/products/`
SugarCRM Inc.	`http://www.sugarcrm.com/crm/products/editions`

B. Complete survey

A printed copy of the entire survey is part of this chapter. Since the online survey uses conditions, some labels are replaced with question codes. Furthermore, the participants of the online version saw just one version of each category, depending on how many products their company offers.

During the last decade, Free and Open Source Software (FOSS) has evolved from a negligible phenomenon to a wide-spread trend. This observation is supported by several studies indicating that the usage of such software has dramatically increased for both business and private users.

The growing economic value of open source software has not gone unnoticed by the industry, resulting in the founding of companies which build their business model solely on open source software.

We are interested in understanding how this development can be commercially sustainable. This survey is trying to determine what open source companies sell and how features of such open source products may look like.

It covers the following aspects:

- Which product features are sold as parts of existing commercial open source software?
- Are some features more important to the customers than others?
- Are there any relations among these features?
- How are these features bundled together?

The survey is split into several sections with each of them containing questions about one of the following areas of interest:

- Legal issues (e.g. licensing)
- Intellectual property
- Support
- Training
- Services

This survey is part of a bachelor thesis undertaken by Florian Weikert and supervised by Prof. Dr. Dirk Riehle, leader of the Open Source Research Group at the Friedrich-Alexander-University of Erlangen-Nuremberg.

Participation in this survey is anonymous and the data is kept strictly confidential. The survey analysis results will be made available to survey participants first and as scientific publications to the general public later.

Thank you very much for your contribution!

There are 36 questions in this survey

Introduction

Please provide some basic information about your company

1 [company_type]What is the dominant business model of your company? *

Please choose **at most** 1 answers:

☐ Software producer: We sell our own software and possibly offer services for it (e.g. MySQL, Jaspersoft)
☐ Service provider: We provide services for specific open source software (e.g. ThoughtWorks, WSO2)
☐ Distributor: We integrate a set of open source software into a final product and offer services (e.g. Red Hat, SUSE)
☐ Other:

2 [products_number]How many different products does your company offer? *

Please write your answer here:

Different versions of one software product count as different products. For example, if your company offers a specific software product in both "Professional" and "Enterprise" edition, please answer this question with "2".

3 [products_names_2]Please assign a unique label to each of the products your company offers. *

Only answer this question if the following conditions are met:
° Answer was 2 at question '2 [products_number]' (How many different products does your company offer?)

Please write your answer(s) here:

Product #1
Product #2

These labels are used to identify each of your products during the survey. For example, whenever a question in this survey aims at the first product, the label you entered for

"Product #1" will be displayed as a reminder. Therefore these labels do not need to match the actual product names.

4 [products_names_3]Please assign a unique label to each of the products your company offers. If your company offers more than three different products, please pick the three most significant ones. *

Only answer this question if the following conditions are met:
° Answer was greater than or equal to 3 at question '2 [products_number]' (How many different products does your company offer?)

Please write your answer(s) here:

Product #1	
Product #2	
Product #3	

These labels are used to identify each of your products during the survey. For example, whenever a question in this survey aims at the first product, the label you entered for "Product #1" will be displayed as a reminder. Therefore these labels do not need to match the actual product names.

Legal Issues

This section contains questions about features concerning legal issues, e.g. licensing.

5 [legal_features_1]Please select for each of the features listed below if they are part of a commercial product your company sells.

Only answer this question if the following conditions are met:
° Answer was 1 at question '2 [products_number]' (How many different products does your company offer?)

Check any that apply:

	This feature is part of our product	Not sure about this feature
Re-branding: Our customers are allowed to replace our trademark with their own	☐	☐
Perpetual licensing: Our clients may receive a license which does not expire	☐	☐
Indemnification: Our company agrees to compensate customers for any damages caused by our products	☐	☐
Warranty: We guarantee specific characteristics of our products to our clients	☐	☐
Non-copyleft usage rights: Customers do not need to publish their own source code when including parts of our code (i.e. avoidance of GPL-like provisions)	☐	☐
Managed release cycles: We assure that new software versions are released at regular intervals	☐	☐
Maintenance: Our customers may receive repairs and bug fixes at no additional cost during a specified period after the purchase	☐	☐
Updates: Our clients are eligible to receive future updates at no additional cost	☐	☐

6 [legal_features_2]Please select for each of the features listed below if they are part of a commercial product your company sells.

Only answer this question if the following conditions are met:
° Answer was 2 at question '2 [products_number]' (How many different products does your company offer?)

Check any that apply:

	{INSERTANS:38367X137X1832SQ001}	{INSERTANS:38367X137X1832SQ002}	Not sure about this feature
Re-branding: Our customers are allowed to replace our trademark with their own	☐	☐	☐
Perpetual licensing: Our clients may receive a license which does not expire	☐	☐	☐
Indemnification: Our company agrees to compensate customers for any damages caused by our products	☐	☐	☐
Warranty: We guarantee specific characteristics of our products to our clients	☐	☐	☐
Non-copyleft usage rights: Customers do not need to publish their own source code when including parts of our code (i.e. avoidance of GPL-like provisions)	☐	☐	☐
Managed release cycles: We assure that new software versions are released at regular intervals	☐	☐	☐
Maintenance: Our customers may receive repairs and bug fixes at no additional cost during a specified period after the purchase	☐	☐	☐
Updates: Our clients are eligible to receive future updates at no additional cost	☐	☐	☐

7 [legal_features_3]Please select for each of the features listed below if they are part of a commercial product your company sells.

Only answer this question if the following conditions are met:
° Answer was greater than or equal to 3 at question '2 [products_number]' (How many different products does your company offer?)

Check any that apply:

	{INSERTANS:38367X137X1845SQ001}	{INSERTANS:38367X137X1845SQ002}	{INSERTANS:38367X137X1845SQ003}	Not sure about this feature
Re-branding: Our customers are allowed to replace our trademark with their own	☐	☐	☐	☐
Perpetual licensing: Our clients may receive a license which does not expire	☐	☐	☐	☐

	{INSERTANS:38367X137X1845SQ001}	{INSERTANS:38367X137X1845SQ002}	{INSERTANS:38367X137X1845SQ003}	Not sure about this feature
Indemnification: Our company agrees to compensate customers for any damages caused by our products	☐	☐	☐	☐
Warranty: We guarantee specific characteristics of our products to our clients	☐	☐	☐	☐
Non-copyleft usage rights: Customers do not need to publish their own source code when including parts of our code (i.e. avoidance of GPL-like provisions)	☐	☐	☐	☐
Managed release cycles: We assure that new software versions are released at regular intervals	☐	☐	☐	☐
Maintenance: Our customers may receive repairs and bug fixes at no additional cost during a specified period after the purchase	☐	☐	☐	☐
Updates: Our clients are eligible to receive future updates at no additional cost	☐	☐	☐	☐

8 [legal_missing]Are there any features in your products which are related to legal issues but which are not mentioned here? Please add a short description, if possible.

Please write your answer here:

9 [legal_ranking]Judging by your experience, how important are the following features to your customers?

Please number each box in order of preference from 1 to 8

☐ Warranty

☐ Indemnification

☐ Non-copyleft usage rights

☐ Managed release cycles

- [ ] Re-branding
- [ ] Perpetual license
- [ ] Updates
- [ ] Maintenance

Proprietary Intellectual Property

Questions in this section aim at features related to proprietary intellectual property of your company, e.g. software and documentation.

10 [ip_features_1]Please select for each of the features listed below if they are part of a commercial product your company sells.

Only answer this question if the following conditions are met:
° Answer was 1 at question '2 [products_number]' (How many different products does your company offer?)

Check any that apply:

	This feature is part of our product	Not sure about this feature
Printed documentation	☐	☐
Digital documentation	☐	☐
Software utilities	☐	☐
Additional functionality (compared to a standard version)	☐	☐
Improved characteristics (compared to a standard version, e.g. performance, security and scalability)	☐	☐
Certification for other software	☐	☐
Certification for hardware	☐	☐
Certification for the use in specific processes (e.g. ISO 9000)	☐	☐
Certification of the development process (e.g. Automotive SPICE)	☐	☐
Availability on physical medium (CD, DVD)	☐	☐
Integrated and configured assembly of several software products (e.g. SUSE Linux)	☐	☐

11 [ip_features_2]Please select for each of the features listed below if they are part of a commercial product your company sells.

Only answer this question if the following conditions are met:
° Answer was 2 at question '2 [products_number]' (How many different products does your company offer?)

Check any that apply:

	{INSERTANS:38367X137X1832SQ001}	{INSERTANS:38367X137X1832SQ002}	Not sure about this feature
Printed documentation	☐	☐	☐
Digital documentation	☐	☐	☐
Software utilities	☐	☐	☐
Additional functionality (compared to a standard version)	☐	☐	☐
Improved characteristics (compared to a standard version, e.g. performance, security and scalability)	☐	☐	☐
Certification for other software	☐	☐	☐
Certification for hardware	☐	☐	☐
Certification for the use in specific processes (e.g. ISO 9000)	☐	☐	☐
Certification of the development process (e.g. Automotive SPICE)	☐	☐	☐
Availability on physical medium (CD, DVD)	☐	☐	☐
Integrated and configured assembly of several software products (e.g. SUSE Linux)	☐	☐	☐

12 [ip_features_3]Please select for each of the features listed below if they are part of a commercial product your company sells.

Only answer this question if the following conditions are met:
° Answer was greater than or equal to 3 at question '2 [products_number]' (How many different products does your company offer?)

Check any that apply:

	{INSERTANS:38367X137X1845SQ001}	{INSERTANS:38367X137X1845SQ002}	{INSERTANS:38367X137X1845SQ003}	Not sure about this feature
Printed documentation	☐	☐	☐	☐
Digital documentation	☐	☐	☐	☐
Software utilities	☐	☐	☐	☐
Additional functionality (compared to a standard version)	☐	☐	☐	☐
Improved characteristics (compared to	☐	☐	☐	☐

	{INSERTANS:38367X137X1845SQ001}	{INSERTANS:38367X137X1845SQ002}	{INSERTANS:38367X137X1845SQ003}	Not sure about this feature
a standard version, e.g. performance, security and scalability)				
Certification for other software	☐	☐	☐	☐
Certification for hardware	☐	☐	☐	☐
Certification for the use in specific processes (e.g. ISO 9000)	☐	☐	☐	☐
Certification of the development process (e.g. Automotive SPICE)	☐	☐	☐	☐
Availability on physical medium (CD, DVD)	☐	☐	☐	☐
Integrated and configured assembly of several software products (e.g. SUSE Linux)	☐	☐	☐	☐

13 [ip_missing]Are there any features related to intellectual property which are not mentioned here? Please add a short description, if possible.

Please write your answer here:

14 [ip_ranking]Judging by your experience, how important are the following features to your customers?

Please number each box in order of preference from 1 to 11

- ☐ Printed documentation
- ☐ Digital documentation
- ☐ Software utilities
- ☐ Additional functionality
- ☐ Improved characteristics
- ☐ Certification for other software
- ☐ Certification for hardware
- ☐ Certification for the use in specific processes
- ☐ Certification of the development process
- ☐ Availability on physical medium
- ☐ Integrated and configured assembly of several software products

Support

This section contains questions about features related to support.

15 [support_features_1]Please select for each of the features listed below if they are part of a commercial product your company sells.

Only answer this question if the following conditions are met:
° Answer was 1 at question '2 [products_number]' (How many different products does your company offer?)

Check any that apply:

	This feature is part of our product	Not sure about this feature
Managed support (Email, Phone, Web Chat)	☐	☐
Unguided support (Forum, Knowledge base, self-help documents)	☐	☐
Guaranteed response time	☐	☐
Guaranteed availability	☐	☐

16 [support_features_2]Please select for each of the features listed below if they are part of a commercial product your company sells.

Only answer this question if the following conditions are met:
° Answer was 2 at question '2 [products_number]' (How many different products does your company offer?)

Check any that apply:

	{INSERTANS:38367X137X1832SQ001}	{INSERTANS:38367X137X1832SQ002}	Not sure about this feature
Managed support (Email, Phone, Web Chat)	☐	☐	☐
Unguided support (Forum, Knowledge base, self-help documents)	☐	☐	☐
Guaranteed response time	☐	☐	☐
Guaranteed availability	☐	☐	☐

17 [support_features_3]Please select for each of the features listed below if they are part of a commercial product your company sells.

Only answer this question if the following conditions are met:
° Answer was greater than or equal to 3 at question '2 [products_number]' (How many different products does your company offer?)

Check any that apply:

	{INSERTANS:38367X137X1845SQ001}	{INSERTANS:38367X137X1845SQ002}	{INSERTANS:38367X137X1845SQ003}	Not sure about this feature
Managed support (Email, Phone, Web Chat)	☐	☐	☐	☐
Unguided support (Forum, Knowledge base, self-help documents)	☐	☐	☐	☐
Guaranteed response time	☐	☐	☐	☐
Guaranteed availability	☐	☐	☐	☐

18 [support_missing]Does your company offer features related to support which are not listed here? Please add a short description, if possible.

Please write your answer here:

19 [support_ranking]Judging by your experience, how important are the following features to your customers?

Please number each box in order of preference from 1 to 4

- [ ] Managed support
- [ ] Unguided support
- [ ] Guaranteed response time
- [ ] Guaranteed availability

Training

This section contains questions about features related to training.

20 [training_features_1]Please select for each of the features listed below if they are part of a commercial product your company sells.

Only answer this question if the following conditions are met:
° Answer was 1 at question '2 [products_number]' (How many different products does your company offer?)

Check any that apply:

	This feature is part of our product	Not sure about this feature
On-site training	☐	☐
Online training (led by an instructor)	☐	☐
Online training (self study)	☐	☐
Professional certification training	☐	☐

21 [training_features_2]Please select for each of the features listed below if they are part of a commercial product your company sells.

Only answer this question if the following conditions are met:
° Answer was 2 at question '2 [products_number]' (How many different products does your company offer?)

Check any that apply:

	{INSERTANS:38367X137X1832SQ001}	{INSERTANS:38367X137X1832SQ002}	Not sure about this feature
On-site training	☐	☐	☐
Online training (led by an instructor)	☐	☐	☐
Online training (self study)	☐	☐	☐
Professional certification training	☐	☐	☐

22 [training_features_3]Please select for each of the features listed below if they are part of a commercial product your company sells.

Only answer this question if the following conditions are met:
° Answer was greater than or equal to 3 at question '2 [products_number]' (How many different products does your company offer?)

Check any that apply:

	{INSERTANS:38367X137X1845SQ001}	{INSERTANS:38367X137X1845SQ002}	{INSERTANS:38367X137X1845SQ003}	Not sure about this feature
On-site training	☐	☐	☐	☐
Online training (led by an instructor)	☐	☐	☐	☐
Online training (self study)	☐	☐	☐	☐
Professional certification training	☐	☐	☐	☐

23 [training_missing]Does your company offer features related to training which are not listed here? Please add a short description, if possible.

Please write your answer here:

24 [training_ranking]Judging by your experience, how important are the following features to your customers?

Please number each box in order of preference from 1 to 4

☐ On-site training

- [ ] Online training (led by an instructor)
- [ ] Online training (self study)
- [ ] Professional certification training

General Services

This section contains questions about services your company may offer in addition to support and training.

25 [gs_features_1]Please select for each of the features listed below if they are part of a commercial product your company sells.

Only answer this question if the following conditions are met:
° Answer was 1 at question '2 [products_number]' (How many different products does your company offer?)

Check any that apply:

	This feature is part of our product	Not sure about this feature
Data migration services: Customers' data is imported from legacy software into our application	☐	☐
Installation & configuration services: We assist our customers in installation and configuration of our software	☐	☐
Integration services: We ensure that our software is well integrated into the customers' business environment (e.g. by programming data connectors)	☐	☐
Hosting services: Our software runs on our own servers ("Software-as-a-Service")	☐	☐
General consulting services: We provide advice about the employment of our software and related issues	☐	☐

26 [gs_features_2]Please select for each of the features listed below if they are part of a commercial product your company sells.

Only answer this question if the following conditions are met:
° Answer was 2 at question '2 [products_number]' (How many different products does your company offer?)

Check any that apply:

	{INSERTANS:38367X137X1832SQ001}	{INSERTANS:38367X137X1832SQ002}	Not sure about this feature
Data migration services: Customers' data is imported from legacy software into our application	☐	☐	☐
Installation & configuration services: We assist our customers in installation and configuration of our software	☐	☐	☐
Integration services: We ensure that our software is well integrated into the customers' business environment (e.g. by programming data connectors)	☐	☐	☐
Hosting services: Our software runs on our own servers ("Software-as-a-Service")	☐	☐	☐
General consulting services: We provide advice about the employment of our software and related issues	☐	☐	☐

27 [gs_features_3]Please select for each of the features listed below if they are part of a commercial product your company sells.

Only answer this question if the following conditions are met:
° Answer was greater than or equal to 3 at question '2 [products_number]' (How many different products does your company offer?)

Check any that apply:

	{INSERTANS:38367X137X1845SQ001}	{INSERTANS:38367X137X1845SQ002}	{INSERTANS:38367X137X1845SQ003}	Not sure about this feature
Data migration services: Customers' data is imported from legacy software into our application	☐	☐	☐	☐
Installation & configuration services: We assist our customers in installation and configuration of our software	☐	☐	☐	☐
Integration services: We ensure that our software	☐	☐	☐	☐

	{INSERTANS:38367X137X1845SQ001}	{INSERTANS:38367X137X1845SQ002}	{INSERTANS:38367X137X1845SQ003}	Not sure about this feature
is well integrated into the customers' business environment (e.g. by programming data connectors)				
Hosting services: Our software runs on our own servers ("Software-as-a-Service")	☐	☐	☐	☐
General consulting services: We provide advice about the employment of our software and related issues	☐	☐	☐	☐

28 [gs_missing]Does your company offer features related to services which are not listed here? Please add a short description, if possible.

Please write your answer here:

29 [gs_ranking]Judging by your experience, how important are the following features to your customers?

Please number each box in order of preference from 1 to 5

- ☐ Data migration services
- ☐ Installation & configuration services
- ☐ Integration services
- ☐ Hosting services
- ☐ General consulting services

Client-specific Services

This section contains questions about services which are specifically tailored to the needs of individual customers.

30 [cs_features_1]Please select for each of the features listed below if they are part of a commercial product your company sells.

Only answer this question if the following conditions are met:
° Answer was 1 at question '2 [products_number]' (How many different products does your company offer?)

Check any that apply:

	This feature is part of our product	Not sure about this feature
Custom implementation: Our customers want us to customize the source code in order to fulfill specific tasks	☐	☐
Client-requested certification: Our clients request a formal guarantee that our products can be used with other software without any problems.	☐	☐

31 [cs_features_2]Please select for each of the features listed below if they are part of a commercial product your company sells.

Only answer this question if the following conditions are met:
° Answer was 2 at question '2 [products_number]' (How many different products does your company offer?)

Check any that apply:

	{INSERTANS:38367X137X1832SQ001}	{INSERTANS:38367X137X1832SQ002}	Not sure about this feature
Custom implementation: Our customers want us to customize the source code in order to fulfill specific tasks	☐	☐	☐
Client-requested certification: Our clients request a formal guarantee that our products can be used with other software without any problems.	☐	☐	☐

32 [cs_features_3]Please select for each of the features listed below if they are part of a commercial product your company sells.

Only answer this question if the following conditions are met:
° Answer was greater than or equal to 3 at question '2 [products_number]' (How many different products does your company offer?)

Check any that apply:

	{INSERTANS:38367X137X1845SQ001}	{INSERTANS:38367X137X1845SQ002}	{INSERTANS:38367X137X1845SQ003}	Not sure about this feature
Custom implementation: Our customers want us to customize the source code in order to fulfill specific tasks	☐	☐	☐	☐
Client-requested certification: Our clients request a formal guarantee that our products can be used with other software without any problems.	☐	☐	☐	☐

33 [cs_missing]Does your company offer features related to client-specific services which are not listed here? Please add a short description, if possible.

Please write your answer here:

34 [cs_ranking]Judging by your experience, how important are the following features to your customers?

Please number each box in order of preference from 1 to 2

- [ ] Custom implementation
- [ ] Client-requested certification

Comments

35 [comment]

We always strive to improve future surveys. Any feedback or criticism will be highly appreciated.

Please write your answer here:

36 [results]Please leave your name and email address if you want to receive a notification about the results of this survey.

Please write your answer(s) here:

Name

Email

Thank you very much for taking the time to fill out this survey.

If you have any questions or feedback feel free to contact me at florian.weikert@informatik.stud.uni-erlangen.de

01.01.1970 – 01:00

Submit your survey.
Thank you for completing this survey.

C. Comments on missing features

- "Apart for designs / branding, we insist code created for one client be released as open source, back to the community." (Service provider on legal features)
- "We license our web gui under cc by/sa/nc to omit freeriders, also enabling scalability with seats deployed" (Software producer on IP related features)
- "Parts of our product are based on proprietary 3rd party offerings, and we license our offerings accordingly, e.g. why should somebody who paid for microsoft gear get our adapter for free." (Software producer on IP related features)
- "Maintenance commitment is usually key, i.e. for how many years will we fix bugs etc." (Software producer on support features)
- "Train the trainer, partner enablement, partner certification." (Software producer on training features)
- "We definitely do NOT host our software, nothing to be unsure about that ..." (Software producer on general services)
- "LPI Certification" (Distributor on training features)
- "The mentioned services are usually provides through system integrators. We provide those services to system integrators in case they don't want to provide them on their own or do not have enough knowledge for a specific task." (Distributor on general service features)
- "Custom implementation and Client-requested certification both do happen, but they are not typical." (Distributor on client-specific service features)

D. Exact numeric values of the rankings

See the following pages for the exact numeric values of frequencies, sample sizes and confidence intervals.

Table D.1.: Frequency of features

	Feature	Total	Software Producer	Service Provider	Distributor
1	Unguided support	100%	100%	100%	100%
2	Managed support	87.5%	83.3%	100%	100%
3	Maintenance	81.5%	83.3%	100%	66.7%
4	Updates	77.8%	77.8%	100%	66.7%
5	On-site training	75%	66.7%	100%	100%
6	Installation and configuration	75%	91.7%	100%	0%
7	Documentation (digital)	70.6%	61.5%	100%	100%
8	Perpetual licensing	70.4%	61.1%	100%	83.3%
9	Warranty	70.4%	77.8%	66.7%	50%
10	Integration	68.8%	83.3%	100%	0%
11	Managed release cycles	66.7%	66.7%	33.3%	83.3%
12	Guaranteed response time	62.5%	58.3%	0%	100%
13	Guaranteed availability	62.5%	58.3%	0%	100%
14	Custom implementation	62.5%	75%	100%	0%
15	Consulting	56.2%	66.7%	100%	0%
16	Additional functionality	52.9%	61.5%	100%	0%
17	Professional certification training	50%	41.7%	0%	100%
18	Documentation (printed)	47.1%	38.5%	0%	100%
19	Data migration	43.8%	50%	100%	0%
20	Improved characteristics	41.2%	46.2%	100%	0%
21	Certification (SW)	41.2%	3%	0%	100%
22	Online training (instructor-led)	37.5%	50%	0%	0%
23	Online training (self-study)	37.5%	41.7%	100%	0%
24	Hosting	37.5%	41.7%	100%	0%
25	Software distribution	35.3%	23.1%	0%	100%
26	Indemnification	33.3%	33.3%	0%	50%
27	Re-branding	29.6%	33.3%	66.7%	0%
28	Software utilities	29.4%	15.4%	0%	100%
29	Certification (HW)	29.4%	3%	0%	33.3%
30	Physical medium	29.4%	15.4%	0%	100%
31	Non-copyleft usage rights	25.9%	38.9%	0%	0%
32	Client-requested certification	25%	33.3%	0%	0%
33	Certification (Processes)	0%	0%	0%	0%
34	Certification (development process)	0%	0%	0%	0%

Table D.2.: Sample sizes for rankings by frequency

	Feature	Total	Software Producer	Service Provider	Distributor
1	Unguided support	16	12	1	3
2	Managed support	16	12	1	3
3	Maintenance	27	18	3	6
4	Updates	27	18	3	6
5	On-site training	16	12	1	3
6	Installation and configuration	16	12	1	3
7	Documentation (digital)	17	13	1	3
8	Perpetual licensing	27	18	3	6
9	Warranty	27	18	3	6
10	Integration	16	12	1	3
11	Managed release cycles	27	18	3	6
12	Guaranteed response time	16	12	1	3
13	Guaranteed availability	16	12	1	3
14	Custom implementation	16	12	1	3
15	Consulting	16	12	1	3
16	Additional functionality	17	13	1	3
17	Professional certification training	16	12	1	3
18	Documentation (printed)	17	13	1	3
19	Data migration	16	12	1	3
20	Improved characteristics	17	13	1	3
21	Certification (SW)	17	13	1	3
22	Online training (instructor-led)	16	12	1	3
23	Online training (self-study)	16	12	1	3
24	Hosting	16	12	1	3
25	Software distribution	17	13	1	3
26	Indemnification	27	18	3	6
27	Re-branding	27	18	3	6
28	Software utilities	17	13	1	3
29	Certification (HW)	17	13	1	3
30	Physical medium	17	13	1	3
31	Non-copyleft usage rights	27	18	3	6
32	Client-requested certification	16	12	1	3
33	Certification (Processes)	17	13	1	3
34	Certification (development process)	17	13	1	3

Table D.3.: Confidence intervals (0.95 level) for frequency of features

	Feature	Total	Software Producer	Service Provider	Distributor
1	Unguided support	0%	0%	0%	0%
2	Managed support	16.21%	21.1%	0%	0%
3	Maintenance	14.65%	17.23%	0%	37.71%
4	Updates	15.68%	19.2%	0%	37.71%
5	On-site training	21.22%	26.67%	0%	0%
6	Installation and configuration	21.22%	15.61%	0%	0%
7	Documentation (digital)	21.66%	26.45%	0%	0%
8	Perpetual licensing	17.22%	22.52%	0%	29.84%
9	Warranty	17.22%	19.2%	53.33%	40.01%
10	Integration	22.7%	21.1%	0%	0%
11	Managed release cycles	17.78%	21.77%	53.33%	29.84%
12	Guaranteed response time	23.72%	27.9%	0%	0%
13	Guaranteed availability	23.72%	27.9%	0%	0%
14	Custom implementation	23.72%	24.5%	0%	0%
15	Consulting	24.31%	26.67%	0%	0%
16	Additional functionality	23.73%	26.45%	0%	0%
17	Professional certification training	24.5%	27.9%	0%	0%
18	Documentation (printed)	23.73%	26.45%	0%	0%
19	Data migration	24.31%	28.29%	0%	0%
20	Improved characteristics	23.4%	27.1%	0%	0%
21	Certification (SW)	23.4%	25.1%	0%	0%
22	Online training (instructor-led)	23.72%	28.29%	0%	0%
23	Online training (self-study)	23.72%	27.9%	0%	0%
24	Hosting	23.72%	27.9%	0%	0%
25	Software distribution	22.72%	22.91%	0%	0%
26	Indemnification	17.78%	21.77%	0%	40.01%
27	Re-branding	17.22%	21.77%	53.33%	0%
28	Software utilities	21.66%	19.62%	0%	0%
29	Certification (HW)	21.66%	25.1%	0%	53.33%
30	Physical medium	21.66%	19.62%	0%	0%
31	Non-copyleft usage rights	16.52%	22.52%	0%	0%
32	Client-requested certification	21.22%	26.67%	0%	0%
33	Certification (Processes)	0%	0%	0%	0%
34	Certification (development process)	0%	0%	0%	0%

Table D.4.: Sample sizes for rankings by participants' opinions

	Feature	Total	Software Producer	Service Provider	Distributor
1	On-site training	8	6	1	1
2	Custom implementation	7	5	1	1
3	Maintenance	13	9	2	2
4	Managed support	8	6	1	1
5	Additional functionality	7	5	1	1
6	Updates	13	9	2	2
7	Integration	8	6	1	1
8	Installation and configuration	8	6	1	1
9	Consulting	8	6	1	1
10	Documentation (digital)	7	5	1	1
11	Improved characteristics	7	5	1	1
12	Guaranteed response time	8	6	1	1
13	Guaranteed availability	8	6	1	1
14	Managed release cycles	13	9	2	2
15	Software utilities	7	5	1	1
16	Certification (SW)	7	5	1	1
17	Online training (instructor-led)	8	6	1	1
18	Software distribution	7	5	1	1
19	Client-requested certification	7	5	1	1
20	Perpetual license	13	9	2	2
21	Certification (HW)	7	5	1	1
22	Re-branding	13	9	2	2
23	Warranty	13	9	2	2
24	Data migration	8	6	1	1
25	Hosting	8	6	1	1
26	Unguided support	8	6	1	1
27	Professional certification training	8	6	1	1
28	Online training (self-study)	8	6	1	1
29	Non-copyleft usage rights	13	9	2	2
30	Indemnification	13	9	2	2
31	Documentation (printed)	7	5	1	1
32	Physical medium	7	5	1	1
33	Certification (development process)	7	5	1	1
34	Certification (Processes)	7	5	1	1

Bibliography

[Aar06] Aardt, Albert van: *Business Models on Open Source Software*. 2006

[Alf10] Alfresco Software Inc.: *Open Source Licensing*. http://wiki.alfresco.com/wiki/Open_Source_Licensing#Do_not_remove_or_alter_our_trademarks. Version: 2010. – Last access on 04.12.2011

[Bon] Bonnabeau, Raymond R.: *Software Licensing Agreements*. http://bssdalaw.com/articles/software_licensing_agreements.pdf. – Last access on 04.12.2011

[Cla07] Classen, H. W.: *A practical guide to software licensing for licensees and licensors: analyses and model forms*. 2nd. American Bar Association, Section of Business Law, 2007 http://books.google.de/books?id=KWYMX61nFj8C&printsec=frontcover&dq=A+practical+guide+for+software+licensing&hl=de&ei=b21rTqGBI4KAOv_4sdUF&sa=X&oi=book_result&ct=result&resnum=1&ved=0CC4Q6AEwAA#v=onepage&q&f=false

[Cle11] Cleeland, Jason: *Using ComfortUpdate*. http://docs.limesurvey.org/Using+ComfortUpdate. Version: 2011. – Last access on 04.12.2011

[CM10] Comino, Stefano ; Manenti, Fabio M.: Dual Licensing in Open Source Software Markets. In: *Source* (2010), Nr. January

[DS07] Doe, Robert B. ; Salyers, Jen C.: Software Licenses. Version: 2nd, 2007. http://www.contextlaw.com/articles/LBCS_Software_Licenses.pdf. In: Andresen, Katheryn A. (Hrsg.): *The Law and Business of Computer Software*. 2nd. West Services, Inc., 2007, Kapitel 18

[Erl00] Erlikh, Len: Leveraging legacy system dollars for e-business. In: *It Professional* 2 (2000), Nr. 3, 17–23. http://dx.doi.org/10.1109/6294.846201. – DOI 10.1109/6294.846201. – ISSN 15209202

[FF00] Feller, Joseph ; Fitzgerald, Brian: A framework analysis of the open source software development paradigm. In: *Proceedings of the twenty first international conference on Information systems* Bd. pp International Conference on Information Systems, Association for Information Systems, 2000, 58–69

[FF02] Feller, Joseph ; Fitzgerald, Brian: *Understanding Open Source Software Development*. Addison-Wesley, 2002 (84). http://portal.acm.org/citation.cfm?id=513726. – ISBN 0201734966

[Fit06] Fitzgerald, Brian: The Transformation of Open Source Software. In: *MIS Quarterly* 30 (2006), Nr. 3, 587–598. http://dx.doi.org/10.1.1.107.2529. – DOI 10.1.1.107.2529. – ISSN 02767783

[Fre07a] Free Software Foundation: *GNU Affero General Public License*. http://www.gnu.org/licenses/agpl.html. Version: 2007. – Last access on 04.12.2011

[Fre07b] Free Software Foundation: *GNU General Public License v3*. http://www.gnu.org/licenses/gpl.html. Version: 2007. – Last access on 04.12.2011

[Hec99] Hecker, Frank: Setting Up Shop: The Business of Open-Source Software. In: *IEEE Software* 16 (1999), Nr. 1, 45–51. http://dx.doi.org/10.1109/52.744568. – DOI 10.1109/52.744568. – ISSN 07407459

[Hel11] Helmreich, Martin: *Best Practices of Adopting Open Source Software in Closed Source Software Products*, Erlangen-Nuremberg, Diss., 2011. http://dirkriehle.com/wp-content/uploads/2011/03/DA-complete.pdf

[KDM04] Kim, C S. ; Djamaludin, I ; Murthy, D N P.: Warranty and discrete preventive maintenance. In: *Reliability Engineering & System Safety* 84 (2004), Juni, Nr. 3, 301–309. http://www.sciencedirect.com/science/article/pii/S0951832003002746. – ISSN 0951–8320

[KGS10] Kumar, Vineet ; Gordon, Brett ; Srinivasan, Kannan: Competitive Strategy for Open Source Software. In: *papersssrncom* (2010). http://papers.ssrn.com/sol3/papers.cfm?abstract_id=1677937

[Kos04] Koskinen, Jussi: *Software maintenance costs*. http://users.jyu.fi/~koskinen/smcosts.htm. Version: 2004. – Last access on 04.12.2011

[Kri03] Krishnamurthy, Sandeep: An Analysis of Open Source Business Models. Version: 2003. http://papers.ssrn.com/sol3/papers.cfm?abstract_id=650001. In: Feller, Joseph (Hrsg.) ; Fitzgerald, Brian (Hrsg.) ; Hissam, Scott A. (Hrsg.) ; Lakhani, Karim R. (Hrsg.) ; University of Washington (Veranst.): *Source* Bd. 54. The MIT Press, 2003, 267–278

[Lau04] Laurent, Andrew M S.: *Understanding open source and free software licensing*. O'Reilly Media, Inc., 2004. – 193 S. http://portal.acm.org/citation.cfm?id=1096111. – ISBN 0596005814

[LT05] Lerner, Josh ; Tirole, Jean: The Economics of Technology Sharing: Open Source and Beyond. In: *Journal of Economic Perspectives* 19 (2005), Nr. 2, 99–120. http://dx.doi.org/10.1257/0895330054048678. – DOI 10.1257/0895330054048678. – ISSN 08953309

[Mar08] Martell, Duncan: *Sun Micro to buy MySQL for $1 billion*. http://www.reuters.com/article/2008/01/16/us-mysql-sunmicrosystems-idUSWNAS661820080116. Version: 2008. – Last access on 22.03.2011

[MWa] Merriam-Webster: *Definition of Support*. http://www.merriam-webster.com/dictionary/support. – Last access on 04.12.2011

[MWb] Merriam-Webster: *Definition of Warranty*. http://www.merriam-webster.com/dictionary/warranty. – Last access on 04.12.2011

[Opea] Open Source Initiative: *Open Source Licenses*. http://www.opensource.org/licenses/index.html. – Last access on 04.12.2011

[Opeb] Open Source Initiative: *The Open Source Definition*. http://www.opensource.org/docs/osd. – Last access on 04.12.2011

[Ope08] Openbravo S.L.U.: *Openbravo: Trademark Guidelines*. http://www.openbravo.com/legal/trademark-guidelines/. Version: 2008. – Last access on 04.12.2011

[OSSa] OSSCube LLC: *About OSSCube*. http://www.osscube.com/about. – Last access on 13.12.2011

[OSSb] OSSCube LLC: *Drupal Development & Drupal Support*. http://www.osscube.com/solutions/drupal/drupal-module-development-support. – Last access on 13.12.2011

[OSSc] OSSCube LLC: *Drupal Development Services*. http://www.osscube.com/solutions/drupal. – Last access on 13.12.2011

[OSSd] OSSCube LLC: *E-Commerce Solutions*. http://www.osscube.com/solutions/e-commerce-solutions. – Last access on 13.12.2011

[OSSe] OSSCube LLC: *Enterprise Portal Development*. http://www.osscube.com/solutions/enterprise-portal-development. – Last access on 13.12.2011

[OSSf] OSSCube LLC: *http://www.osscube.com/training*. http://www.osscube.com/training. – Last access on 13.12.2011

[OSSg] OSSCube LLC: *Moodle Development Services*. http://www.osscube.com/solutions/moodle. – Last access on 13.12.2011

[OSSh] OSSCube LLC: *MySQL Migration*. http://www.osscube.com/technology/mysql/migration. – Last access on 13.12.2011

[OSSi] OSSCube LLC: *MySQL Performance Tuning*. http://www.osscube.com/technology/mysql/performance-tuning. – Last access on 13.12.2011

[OSSj] OSSCube LLC: *Open Source Products*. http://www.osscube.com/services/open-source-products. – Last access on 13.12.2011

[OSSk] OSSCube LLC: *Open Source Services*. http://www.osscube.com/services. – Last access on 13.12.2011

[OSSl] OSSCube LLC: *PHP Certification Exam Training by OSSCube*. http://www.osscube.com/training/zend/php-certification. – Last access on 13.12.2011

[OSSm] OSSCube LLC: *SugarCRM Services*. http://www.osscube.com/solutions/sugarcrm/lead-management-crm. – Last access on 13.12.2011

[OSSn] OSSCube LLC: *SugarCRM Services and Products from Authorized SugarCRM Gold Partner*. http://www.osscube.com/solutions/sugarcrm. – Last access on 13.12.2011

[PG08] Pettey, Christy ; Goasduff, Laurence: *Gartner Highlights Key Predictions for IT Organisations and Users in 2008 and Beyond*. http://www.gartner.com/it/page.jsp?id=593207. Version: 2008. – Last access on 04.12.2011

[PS11] Pettey, Christy ; Stevens, Holly: *Gartner Says Sales of Mobile Devices Grew 5.6 Percent in Third Quarter of 2011; Smartphone Sales Increased 42 Percent*. http://www.gartner.com/it/page.jsp?id=1848514. Version: 2011. – Last access on 04.12.2011

[Reda] Red Hat Inc.: *Add-Ons for Red Hat Enterprise Linux*. http://www.redhat.com/rhel/add-ons/. – Last access on 13.12.2011

[Redb] Red Hat Inc.: *Certifications & Accreditations*. http://www.redhat.com/solutions/government/certifications/. – Last access on 13.12.2011

[Redc] Red Hat Inc.: *Custom Engineering Services*. http://www.redhat.com/services/custom/benefits/. – Last access on 13.12.2011

[Redd] Red Hat Inc.: *Cygwin*. http://www.redhat.com/services/custom/cygwin/. – Last access on 13.12.2011

[Rede] Red Hat Inc.: *Enterprise Linux or Fedora?* http://www.redhat.com/software/rhelorfedora/. – Last access on 13.12.2011

[Redf] Red Hat Inc.: *Global Support Services*. https://access.redhat.com/support/. – Last access on 13.12.2011

[Redg] Red Hat Inc.: *Hardware Certification*. https://access.redhat.com/support/offerings/hwcert/. – Last access on 13.12.2011

[Redh] Red Hat Inc.: *JBoss Enterprise Middleware*. http://www.redhat.com/jboss/. – Last access on 13.12.2011

[Redi] Red Hat Inc.: *Production Support Service Level Agreement*. https://access.redhat.com/support/offerings/production/sla.html. – Last access on 13.12.2011

[Redj] Red Hat Inc.: *Purchasing Red Hat Enterprise Linux*. http://www.redhat.com/rhel/purchasing_guide.html. – Last access on 13.12.2011

[Redk] Red Hat Inc.: *Recently Supported Applications on Red Hat Enterprise Linux 6*. http://www.redhat.com/rhel/compatibility/software/. – Last access on 13.12.2011

[Redl] Red Hat Inc.: *Red Hat Cloud*. http://www.redhat.com/solutions/cloud/. – Last access on 13.12.2011

[Redm] Red Hat Inc.: *Red Hat Consulting - Services*. http://www.redhat.com/consulting/services/. – Last access on 13.12.2011

[Redn] Red Hat Inc.: *Red Hat Consulting - Solutions*. http://www.redhat.com/consulting/solutions/. – Last access on 13.12.2011

[Redo] Red Hat Inc.: *Red Hat Enterprise Agreement*. http://www.redhat.com/licenses/us.html. – Last access on 13.12.2011

[Redp] Red Hat Inc.: *Red Hat Enterprise Linux*. http://www.redhat.com/rhel/. – Last access on 13.12.2011

[Redq] Red Hat Inc.: *Red Hat Enterprise Linux*. http://fedoraproject.org/wiki/Red_Hat_Enterprise_Linux#Is_Red_Hat_Enterprise_Linux_a_open_source_product.3F. – Last access on 13.12.2011

[Redr] Red Hat Inc.: *Red Hat Enterprise Linux*. https://access.redhat.com/knowledge/docs/Red_Hat_Enterprise_Linux/. – Last access on 13.12.2011

[Reds] Red Hat Inc.: *Red Hat Enterprise Linux Technology capabilities and limits*. http://www.redhat.com/rhel/compare/. – Last access on 13.12.2011

[Redt] Red Hat Inc.: *Red Hat Hosting Program* . http://www.redhat.com/partners/hostingpartners/. – Last access on 13.12.2011

[Redu] Red Hat Inc.: *Red Hat Open Source Assurance FAQ*. http://www.redhat.com/software/assurance/faq.html. – Last access on 13.12.2011

[Redv] Red Hat Inc.: *redhat.com | Certified Hardware*. https://hardware.redhat.com/page.cgi?id=faq.html. – Last access on 13.12.2011

[Redw] Red Hat Inc.: *Technical Account Management*. https://access.redhat.com/support/offerings/tam/. – Last access on 13.12.2011

[Redx] Red Hat Inc.: *Training and Certification*. http://www.redhat.com/training/. – Last access on 13.12.2011

[Redy] Red Hat Inc.: *Why Red Hat?* http://www.redhat.com/why_red_hat/. – Last access on 13.12.2011

[RI09] Riehle, Dirk ; Interview partner 3: *Confidential Interview 3*. 2009

[RI10a] Riehle, Dirk ; Interview partner 1: *Confidential Interview 1*. 2010

[RI10b] Riehle, Dirk ; Interview partner 2: *Confidential Interview 2*. 2010

[RI10c] Riehle, Dirk ; Interview partner 3: *Confidential Interview 4*. 2010

[Rie07] Riehle, Dirk: The Economic Motivation of Open Source Software: Stakeholder Perspectives. In: *Computer* 40 (2007), Nr. 4, 25–32. http://dx.doi.org/10.1109/MC.2007.147. – DOI 10.1109/MC.2007.147. – ISSN 00189162

[Rie09] Riehle, Dirk: The Commercial Open Source Business Model. In: *Proceedings of the Fifteenth Americas Conference on Information Systems* Bd. AMCIS 2009, 2009, 1–10

[Rie10a] Riehle, Dirk: Single Vendor Open Source. In: *(Unpublished) Script of the lecture "Agile Methods and Open Source Software Development" in the winter term 2009/2010 at the University of Erlangen-Nuremburg* Lecture 13 (2010)

[Rie10b] Riehle, Dirk: The single-vendor commercial open course business model. In: *Information Systems and EBusiness Management* (2010), 1–13. http://dx.doi.org/10.1007/s10257-010-0149-x. – DOI 10.1007/s10257–010–0149–x. – ISSN 16179846

[Rie11] Riehle, Dirk: Controlling and Steering Open Source Projects. In: *IEEE Computer Society* (2011), Nr. July. http://dirkriehle.com/2011/07/18/controlling-and-steering-open-source-projects/

[RRS09] Rossi, Bruno ; Russo, Barbara ; Succi, Giancarlo: Analysis of Open Source Software Development Iterations by Means of Burst Detection Techniques. In: Boldyreff, Cornelia (Hrsg.) ; Crowston, Kevin (Hrsg.) ; Lundell, Björn (Hrsg.) ; Wasserman, Anthony I. (Hrsg.): *Open Source Ecosystems: Diverse Communities Interacting* Bd. 299, Springer Berlin Heidelberg, 2009 (IFIP Advances in Information and Communication Technology). – ISBN 9783642020315, 83–93

[Som05] Sommerville, Ian: Software Documentation. Version: 2005. `http://www.google.com/patents?hl=en&lr=&vid=USPAT5903897&id=6WEXAAAAEBAJ&oi=fnd&dq=Software+Documentation&printsec=abstract`. In: *Software Engineering Volume 2 The Supporting Process*. 2005, 143–154

[Suga] SugarCRM Inc.: *http://www.sugarcrm.com/crm/about/press-releases/20041025-sugarcrm-2.0.html*. `SugarCRM2.0ReleaseCompletesCoreSFAFeatureSet`. – Last access on 13.12.2011

[Sugb] SugarCRM Inc.: *http://www.sugarcrm.com/crm/support/offerings*. `SupportOfferings`. – Last access on 13.12.2011

[Sugc] SugarCRM Inc.: *Master Subscription Agreement*. `http://www.sugarcrm.com/crm/master-subscription-agreement`. – Last access on 13.12.2011

[Sugd] SugarCRM Inc.: *Professional Services*. `http://www.sugarcrm.com/crm/services/professional-services.html`. – Last access on 13.12.2011

[Suge] SugarCRM Inc.: *Solution Components Sugar 6.3 Release*. `http://media.sugarcrm.com/datasheets/EditionsComparison_CE.pdf`. – Last access on 13.12.2011

[Sugf] SugarCRM Inc.: *Sugar Documentation*. `http://www.sugarcrm.com/crm/support/documentation`. – Last access on 13.12.2011

[Sugg] SugarCRM Inc.: *Sugar Subscriptions & Pricing*. `http://www.sugarcrm.com/crm/products/editions`. – Last access on 13.12.2011

[Sugh] SugarCRM Inc.: *Sugar Ultimate Documentation*. `http://www.sugarcrm.com/crm/support/documentation/SugarUltimate`. – Last access on 13.12.2011

[Sugi] SugarCRM Inc.: *Sugar University*. `http://www.sugarcrm.com/university/`. – Last access on 13.12.2011

[Sugj] SugarCRM Inc.: *Sugar University FAQs*. `http://www.sugarcrm.com/university/faqs.php`. – Last access on 13.12.2011

[Sugk] SugarCRM Inc.: *SugarCRM Reseller Levels and Benefits*. http://www.sugarcrm.com/crm/partners/benefits.html. – Last access on 13.12.2011

[Sugl] SugarCRM Inc.: *SugarCRM's Sugar Community Edition License*. http://www.sugarcrm.com/crm/gplv3-faq.html. – Last access on 13.12.2011

[Sugm] SugarCRM Inc.: *Support*. http://www.sugarcrm.com/crm/support. – Last access on 13.12.2011

[Sugn] SugarCRM Inc.: *Tools & Tutorials*. http://developers.sugarcrm.com/tutorials.php?id=recent. – Last access on 13.12.2011

[VÖ03] Välimäki, Mikko: Dual Licensing in Open Source Software Industry. In: *Business* 8 (2003), Nr. 1, 63–75. http://papers.ssrn.com/sol3/papers.cfm?abstract_id=1261644

[VÖ05] Välimäki, Mikko: *The Rise of Open Source Licensing A Challenge to the Use of Intellectual Property in the So ware Industry*. Turre Publishing, 2005 http://lib.tkk.fi/Diss/2005/isbn9529187793/. ISSN 87566222

[Wei11] Weikert, Florian: *How To Earn Money With Open Source Software*, University of Erlangen-Nuremberg, (Unpublished) paper, 2011

[WH05] Weinstock, Charles B. ; Hissam, Scott A.: Making Lightening Strike Twice. In: *Perspectives on Free and Open Source Software*. MIT Press, 2005, S. pp. 143–160.